

“This book helped me solve a Windows networking problem that I had been struggling with for a long time. This book is a must-have for anyone who is trying to implement and maintain a Microsoft file and print server environment—that even includes environments without Samba!”

—*Brad Jones, Senior Network/Systems Architect*

“The book is outstanding. The sections on printing are THE BEST I have ever seen—totally excellent. This book tells network admins the how-and-why of subjects that no one else can cover like [the authors]. In plain language, the authors tell you how to properly configure the various sections to get it all working. [The authors] really, really understand what an admin is going through and have combined that with their own global experience to make a smashingly great book. A must-have manual to use and learn from.”

—*Ismet Kursunoglu, MD, FCCP, Medical Director, Orbitonix*

“This book is for anyone looking to implement Samba for the first time, upgrade from an existing version, or even streamline an existing installation—in any type of environment. It is a much-needed, comprehensive reference. I would feel confident going into any environment to implement a Samba solution armed with this book. Written by the authors of the Samba software, the book’s first-hand perspective gives it an edge in dealing with the new features present in the updated release.”

—*Steve Elgersma, Systems Administrator, Princeton University,
Department of Computer Science*

“[This book contains] some absolutely outstanding documentation. It is broken into self-contained chapters that start by laying out how a certain task or protocol works in general, and then how to configure Samba to take part in it. Considering that Samba can perform so many different roles, the mix-and-match method is a lot more sensible. Even if you don’t use Samba, consider [this book] as a reference for troubleshooting Windows problems—I’ve found [this book offers] a far more complete and focused discussion of Windows technologies for the sysadmin than any Microsoft book or webpage.”

—*Kristopher Magnusson, Open Source Software Advocate*

“When I showed some Microsoft network admins I have been working with *The Official Samba-3 HOWTO and Reference Guide*, they were more than just impressed. Good work on good docs.”

—*C. Lee Taylor, National IT/IS Manager for
Scania South Africa (Pty.) Ltd.*

BRUCE PERENS' OPEN SOURCE SERIES

- ◆ *Managing Linux Systems with Webmin: System Administration and Module Development*
Jamie Cameron
- ◆ *Understanding the Linux Virtual Memory Manager*
Mel Gorman
- ◆ *Implementing CIFS: The Common Internet File System*
Christopher R. Hertel
- ◆ *Embedded Software Development with eCos*
Anthony J. Massa
- ◆ *Rapid Application Development with Mozilla*
Nigel McFarlane
- ◆ *The Linux Development Platform: Configuring, Using, and Maintaining a Complete Programming Environment*
Rafeeq Ur Rehman, Christopher Paul
- ◆ *Intrusion Detection Systems with Snort: Advanced IDS Techniques with Snort, Apache, MySQL, PHP, and ACID*
Rafeeq Ur Rehman
- ◆ *The Official Samba-3 HOWTO and Reference Guide*
John H. Terpstra, Jelmer R. Vernooij, Editors

The Official Samba-3 HOWTO and Reference Guide

*John H. Terpstra
and
Jelmer R. Vernooij,
Editors*



Prentice Hall PTR
Upper Saddle River, New Jersey 07458
www.phptr.com

Library of Congress Cataloging-in-Publication Data

A CIP catalog record for this book can be obtained from the Library of Congress.

Editorial/production supervision: *Mary Sudul*

Cover design director: *Jerry Votta*

Cover design: *DesignSource*

Manufacturing manager: *Maura Zaldivar*

Acquisitions editor: *Jill Harry*

Editorial assistant: *Brenda Mulligan*

Marketing manager: *Dan DePasquale*



© 2004 John H. Terpstra

Published by Prentice Hall PTR

Upper Saddle River, New Jersey 07458

This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

Prentice Hall books are widely used by corporations and government agencies for training, marketing, and resale.

The publisher offers discounts on this book when ordered in bulk quantities. For more information, contact Corporate Sales Department, Phone: 800-382-3419; FAX: 201-236-7141;

E-mail: corpsales@prenhall.com

Or write: Prentice Hall PTR, Corporate Sales Dept., One Lake Street, Upper Saddle River, NJ 07458.

Printed in the United States of America

2nd Printing

ISBN 0-13-145355-6

Pearson Education LTD.

Pearson Education Australia PTY, Limited

Pearson Education Singapore, Pte. Ltd.

Pearson Education North Asia Ltd.

Pearson Education Canada, Ltd.

Pearson Educación de Mexico, S.A. de C.V.

Pearson Education — Japan

Pearson Education Malaysia, Pte. Ltd.

ATTRIBUTION

How to Install and Test SAMBA

- Andrew Tridgell <tridge@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Karl Auer <kauer@biplane.com.au>
- Dan Shearer <dan@samba.org>

Fast Start: Cure for Impatience

- John H. Terpstra <jht@samba.org>

Server Types and Security Modes

- Andrew Tridgell <tridge@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>

Domain Control

- John H. Terpstra <jht@samba.org>
- Gerald (Jerry) Carter <jerry@samba.org>
- David Bannon <dbannon@samba.org>
- Guenther Deschner <gd@suse.de> (LDAP updates)

Backup Domain Control

- John H. Terpstra <jht@samba.org>
- Volker Lendecke <Volker.Lendecke@SerNet.DE>
- Guenther Deschner <gd@suse.de> (LDAP updates)

Domain Membership

- John H. Terpstra <jht@samba.org>

- Jeremy Allison <jra@samba.org>
- Gerald (Jerry) Carter <jerry@samba.org>
- Andrew Tridge <tridge@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- Guenther Deschner <gd@suse.de> (LDAP updates)

Stand-alone Servers

- John H. Terpstra <jht@samba.org>

MS Windows Network Configuration Guide

- John H. Terpstra <jht@samba.org>

Network Browsing

- John H. Terpstra <jht@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>

Account Information Databases

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Gerald (Jerry) Carter <jerry@samba.org>
- Jeremy Allison <jra@samba.org>
- Guenther Deschner <gd@suse.de> (LDAP updates)
- Olivier (lem) Lemaire <olem@IDEALX.org>

Group Mapping MS Windows and UNIX

- John H. Terpstra <jht@samba.org>
- Jean François Micouleau
- Gerald (Jerry) Carter <jerry@samba.org>

File, Directory and Share Access Controls

- John H. Terpstra <jht@samba.org>
- Jeremy Allison <jra@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org> (drawing)

File and Record Locking

- Jeremy Allison <jra@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Eric Roseme <eric.roseme@hp.com>

Securing Samba

- Andrew Tridgell <tridge@samba.org>
- John H. Terpstra <jht@samba.org>

Interdomain Trust Relationships

- John H. Terpstra <jht@samba.org>
- Rafal Szczesniak <mimir@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org> (drawing)
- Stephen Langasek <vorlon@netexpress.net>

Hosting a Microsoft Distributed File System Tree

- Shirish Kalele <samba@samba.org>
- John H. Terpstra <jht@samba.org>

Classical Printing Support

- Kurt Pfeifle <kpfeifle@danka.de>
- Gerald (Jerry) Carter <jerry@samba.org>
- John H. Terpstra <jht@samba.org>

CUPS Printing Support

- Kurt Pfeifle <kpfeifle@danka.de>
- Ciprian Vizitiu <CVizitiu@gbif.org> (drawings)
- Jelmer R. Vernooij <jelmer@samba.org> (drawings)

Stackable VFS modules

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Tim Potter <tpot@samba.org>
- Simo Sorce (original vfs_skel README)
- Alexander Bokovoy (original vfs_netatalk docs)
- Stefan Metzmacher (Update for multiple modules)

Winbind: Use of Domain Accounts

- Tim Potter <tpot@linuxcare.com.au>
- Andrew Tridgell <tridge@samba.org>
- Naag Mummaneni <getnag@rediffmail.com> (Notes for Solaris)
- John Trostel <jtrostel@snapserver.com>
- Jelmer R. Vernooij <jelmer@samba.org>

- John H. Terpstra <jht@samba.org>

Advanced Network Management

- John H. Terpstra <jht@samba.org>

System and Account Policies

- John H. Terpstra <jht@samba.org>

Desktop Profile Management

- John H. Terpstra <jht@samba.org>

PAM-Based Distributed Authentication

- John H. Terpstra <jht@samba.org>
- Stephen Langasek <vorlon@netexpress.net>

Integrating MS Windows Networks with Samba

- John H. Terpstra <jht@samba.org>

Unicode/Charsets

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- TAKAHASHI Motonobu <monyo@home.monyo.com> (Japanese character support)

Backup Techniques

- John H. Terpstra <jht@samba.org>

High Availability

- John H. Terpstra <jht@samba.org>
- Jeremy Allison <jra@samba.org>

Upgrading from Samba-2.x to Samba-3.0.0

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Gerald (Jerry) Carter <jerry@samba.org>

Migration from NT4 PDC to Samba-3 PDC

- John H. Terpstra <jht@samba.org>

SWAT The Samba Web Administration Tool

- John H. Terpstra <jht@samba.org>

The Samba Checklist

- Andrew Tridgell <tridge@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>

- Dan Shearer <dan@samba.org>

Analyzing and Solving Samba Problems

- Gerald (Jerry) Carter <jerry@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- David Bannon <dbannon@samba.org>
- Dan Shearer <dan@samba.org>

Reporting Bugs

- John H. Terpstra <jht@samba.org>
- Jelmer R. Vernooij <jelmer@samba.org>
- Andrew Tridgell <tridge@samba.org>

How to Compile Samba

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Andrew Tridgell <tridge@samba.org>

Portability

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>

Samba and Other CIFS Clients

- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>
- Dan Shearer <dan@samba.org>
- Jim McDonough <jmcd@us.ibm.com> (OS/2)

Samba Performance Tuning

- Paul Cochrane <paulc@dt.h.scot.nhs.uk>
- Jelmer R. Vernooij <jelmer@samba.org>
- John H. Terpstra <jht@samba.org>

DNS and DHCP Configuration Guide

- John H. Terpstra <jht@samba.org>

ABSTRACT

The editors wish to thank you for your decision to purchase this book. The Official Samba-3 HOWTO and Reference Guide is the result of many years of accumulation of information, feedback, tips, hints, and happy solutions.

Please note that this book is a living document, the contents of which are constantly being updated. We encourage you to contribute your tips, techniques, helpful hints, and your special insight into the Windows networking world to help make the next generation of this book even more valuable to Samba users.

We have made a concerted effort to document more comprehensively than has been done previously the information that may help you to better deploy Samba and to gain more contented network users.

This book provides example configurations, it documents key aspects of Microsoft Windows networking, provides in-depth insight into the important configuration of Samba-3, and helps to put all of these into a useful framework.

The most recent electronic versions of this document can be found at <http://www.samba.org/> on the “*Documentation*” page.

Updates, patches and corrections are most welcome. Please email your contributions to any one of the following:

Jelmer Vernooij (jelmer@samba.org)
John H. Terpstra (jht@samba.org)
Gerald (Jerry) Carter (jerry@samba.org)

We wish to advise that only original and unencumbered material can be published. Please do not submit content that is not your own work unless proof of consent from the copyright holder accompanies your submission.

FOREWORD

Over the last few years, the Samba project has undergone a major transformation. From a small project used only by people who dream in machine code, Samba has grown to be an integral part of the IT infrastructure of many businesses. Along with the growth in the popularity of Samba there has been a corresponding growth in the ways that it can be used, and a similar growth in the number of configuration options and the interactions between them.

To address this increasing complexity a wealth of documentation has been written on Samba, including numerous HOWTOs, diagnostic tips, manual pages, and explanations of important pieces of technology that Samba relies on. While it has been gratifying to see so much documentation being written, the sheer volume of different types of documentation has proved difficult to navigate, thus reducing its value to system administrators trying to cope with the complexity.

This book gathers together that wealth of information into a much more accessible form, to allow system administrators to quickly find what they need. The breadth of technical information provided ensures that even the most demanding administrators will find something helpful.

I am delighted that the Samba documentation has now developed to the extent that it can be presented usefully as a book, and I am grateful for the efforts of the many people who have contributed so much toward this result. Enjoy!

Andrew Tridgell
President, Samba Team
July 2003

CONTENTS

Contents

ATTRIBUTION	v
ABSTRACT	xi
FOREWORD	xiii
LIST OF FIGURES	xxxv
LIST OF TABLES	xxxvii
Part I General Installation	xxxvii
PREFACE AND INTRODUCTION	1
Chapter 1 HOW TO INSTALL AND TEST SAMBA	5
1.1 Obtaining and Installing Samba	5
1.2 Configuring Samba (smb.conf)	5
1.2.1 Configuration file syntax	5
1.2.2 Example Configuration	6
1.2.2.1 Test Your Config File with testparm	6
1.2.3 SWAT	7
1.3 List Shares Available on the Server	7
1.4 Connect with a UNIX Client	7
1.5 Connect from a Remote SMB Client	8
1.6 What If Things Don't Work?	8
1.7 Common Errors	8
1.7.1 Large Number of smbd Processes	9
1.7.2 Error Message: open_oplock_ipc	9
1.7.3 <i>"The network name cannot be found"</i>	9
Chapter 2 FAST START: CURE FOR IMPATIENCE	11
2.1 Features and Benefits	11
2.2 Description of Example Sites	11
2.3 Worked Examples	12
	xv

2.3.1	Stand-alone Server	12
2.3.1.1	Anonymous Read-Only Document Server	12
2.3.1.2	Anonymous Read-Write Document Server	14
2.3.1.3	Anonymous Print Server	14
2.3.1.4	Secure Read-Write File and Print Server	16
2.3.2	Domain Member Server	19
2.3.2.1	Example Configuration	20
2.3.3	Domain Controller	23
2.3.3.1	Example: Engineering Office	24
2.3.3.2	A Big Organization	26
Part II Server Configuration Basics		29
Chapter 3 SERVER TYPES AND SECURITY MODES		31
3.1	Features and Benefits	31
3.2	Server Types	32
3.3	Samba Security Modes	32
3.3.1	User Level Security	33
3.3.1.1	Example Configuration	33
3.3.2	Share Level Security	33
3.3.2.1	Example Configuration	34
3.3.3	Domain Security Mode (User Level Security)	34
3.3.3.1	Example Configuration	34
3.3.4	ADS Security Mode (User Level Security)	35
3.3.4.1	Example Configuration	35
3.3.5	Server Security (User Level Security)	36
3.3.5.1	Example Configuration	37
3.4	Password Checking	37
3.5	Common Errors	38
3.5.1	What Makes Samba a Server?	39
3.5.2	What Makes Samba a Domain Controller?	39
3.5.3	What Makes Samba a Domain Member?	39
3.5.4	Constantly Losing Connections to Password Server	39
Chapter 4 DOMAIN CONTROL		41
4.1	Features and Benefits	42
4.2	Basics of Domain Control	44
4.2.1	Domain Controller Types	44
4.2.2	Preparing for Domain Control	45
4.3	Domain Control — Example Configuration	47
4.4	Samba ADS Domain Control	49
4.5	Domain and Network Logon Configuration	50
4.5.1	Domain Network Logon Service	50
4.5.1.1	Example Configuration	50
4.5.1.2	The Special Case of MS Windows XP Home Edition	50
4.5.1.3	The Special Case of Windows 9x/Me	51
4.5.2	Security Mode and Master Browsers	52

4.6	Common Errors	53
4.6.1	“\$” Cannot Be Included in Machine Name	53
4.6.2	Joining Domain Fails Because of Existing Machine Account	54
4.6.3	The System Cannot Log You On (C000019B)	54
4.6.4	The Machine Trust Account Is Not Accessible	55
4.6.5	Account Disabled	55
4.6.6	Domain Controller Unavailable	55
4.6.7	Cannot Log onto Domain Member Workstation After Joining Domain	55
Chapter 5	BACKUP DOMAIN CONTROL	57
5.1	Features and Benefits	57
5.2	Essential Background Information	58
5.2.1	MS Windows NT4-style Domain Control	59
5.2.1.1	Example PDC Configuration	60
5.2.2	LDAP Configuration Notes	60
5.2.3	Active Directory Domain Control	61
5.2.4	What Qualifies a Domain Controller on the Network?	62
5.2.5	How does a Workstation find its Domain Controller?	62
5.2.5.1	NetBIOS Over TCP/IP Enabled	62
5.2.5.2	NetBIOS Over TCP/IP Disabled	62
5.3	Backup Domain Controller Configuration	62
5.3.1	Example Configuration	63
5.4	Common Errors	64
5.4.1	Machine Accounts Keep Expiring	64
5.4.2	Can Samba Be a Backup Domain Controller to an NT4 PDC?	65
5.4.3	How Do I Replicate the smbpasswd File?	65
5.4.4	Can I Do This All with LDAP?	65
Chapter 6	DOMAIN MEMBERSHIP	67
6.1	Features and Benefits	67
6.2	MS Windows Workstation/Server Machine Trust Accounts	68
6.2.1	Manual Creation of Machine Trust Accounts	69
6.2.2	Managing Domain Machine Accounts using NT4 Server Manager	70
6.2.3	On-the-Fly Creation of Machine Trust Accounts	71
6.2.4	Making an MS Windows Workstation or Server a Domain Member	71
6.2.4.1	Windows 200x/XP Professional Client	71
6.2.4.2	Windows NT4 Client	71
6.2.4.3	Samba Client	72
6.3	Domain Member Server	72
6.3.1	Joining an NT4-type Domain with Samba-3	72
6.3.2	Why Is This Better Than security = server?	74
6.4	Samba ADS Domain Membership	75
6.4.1	Configure smb.conf	75
6.4.2	Configure /etc/krb5.conf	76
6.4.3	Create the Computer Account	77
6.4.3.1	Possible Errors	78
6.4.4	Testing Server Setup	78
6.4.5	Testing with smbclient	78

6.4.6	Notes	78
6.5	Sharing User ID Mappings between Samba Domain Members	79
6.6	Common Errors	79
6.6.1	Cannot Add Machine Back to Domain	79
6.6.2	Adding Machine to Domain Fails	80
6.6.3	I Can't Join a Windows 2003 PDC	80
Chapter 7	STAND-ALONE SERVERS	81
7.1	Features and Benefits	81
7.2	Background	81
7.3	Example Configuration	82
7.3.1	Reference Documentation Server	82
7.3.2	Central Print Serving	83
7.4	Common Errors	84
Chapter 8	MS WINDOWS NETWORK CONFIGURATION GUIDE	85
8.1	Features and Benefits	85
8.2	Technical Details	85
8.2.1	TCP/IP Configuration	85
8.2.1.1	MS Windows XP Professional	86
8.2.1.2	MS Windows 2000	87
8.2.1.3	MS Windows Me	89
8.2.2	Joining a Domain: Windows 2000/XP Professional	91
8.2.3	Domain Logon Configuration: Windows 9x/Me	93
8.3	Common Errors	94
Part III	Advanced Configuration	101
Chapter 9	NETWORK BROWSING	103
9.1	Features and Benefits	103
9.2	What Is Browsing?	104
9.3	Discussion	104
9.3.1	NetBIOS over TCP/IP	105
9.3.2	TCP/IP without NetBIOS	105
9.3.3	DNS and Active Directory	106
9.4	How Browsing Functions	107
9.4.1	Configuring WORKGROUP Browsing	108
9.4.2	DOMAIN Browsing Configuration	109
9.4.3	Forcing Samba to Be the Master	110
9.4.4	Making Samba the Domain Master	111
9.4.5	Note about Broadcast Addresses	111
9.4.6	Multiple Interfaces	111
9.4.7	Use of the Remote Announce Parameter	112
9.4.8	Use of the Remote Browse Sync Parameter	112
9.5	WINS — The Windows Internetworking Name Server	112
9.5.1	WINS Server Configuration	113
9.5.2	WINS Replication	114

9.5.3	Static WINS Entries	114
9.6	Helpful Hints	115
9.6.1	Windows Networking Protocols	115
9.6.2	Name Resolution Order	116
9.7	Technical Overview of Browsing	116
9.7.1	Browsing Support in Samba	117
9.7.2	Problem Resolution	118
9.7.3	Cross-Subnet Browsing	118
9.7.3.1	Behavior of Cross-Subnet Browsing	118
9.8	Common Errors	121
9.8.1	How Can One Flush the Samba NetBIOS Name Cache without Restarting Samba?	122
9.8.2	Server Resources Can Not Be Listed	122
9.8.3	I get an 'Unable to browse the network' error	122
9.8.4	Browsing of Shares and Directories is Very Slow	122
Chapter 10	ACCOUNT INFORMATION DATABASES	125
10.1	Features and Benefits	125
10.1.1	Backward Compatibility Backends	125
10.1.2	New Backends	126
10.2	Technical Information	127
10.2.1	Important Notes About Security	128
10.2.1.1	Advantages of Encrypted Passwords	129
10.2.1.2	Advantages of Non-Encrypted Passwords	130
10.2.2	Mapping User Identifiers between MS Windows and UNIX	130
10.2.3	Mapping Common UIDs/GIDs on Distributed Machines	130
10.3	Account Management Tools	131
10.3.1	The <i>smbpasswd</i> Command	131
10.3.2	The <i>pdbedit</i> Command	132
10.4	Password Backends	133
10.4.1	Plaintext	133
10.4.2	smbpasswd — Encrypted Password Database	134
10.4.3	tdbsam	134
10.4.4	ldapsam	134
10.4.4.1	Supported LDAP Servers	135
10.4.4.2	Schema and Relationship to the RFC 2307 posixAccount	135
10.4.4.3	OpenLDAP Configuration	136
10.4.4.4	Initialize the LDAP Database	137
10.4.4.5	Configuring Samba	139
10.4.4.6	Accounts and Groups Management	139
10.4.4.7	Security and sambaSamAccount	141
10.4.4.8	LDAP Special Attributes for sambaSamAccounts	141
10.4.4.9	Example LDIF Entries for a sambaSamAccount	142
10.4.4.10	Password Synchronization	143
10.4.5	MySQL	143
10.4.5.1	Creating the Database	143
10.4.5.2	Configuring	143
10.4.5.3	Using Plaintext Passwords or Encrypted Password	144

10.4.5.4	Getting Non-Column Data from the Table	145
10.4.6	XML	145
10.5	Common Errors	145
10.5.1	Users Cannot Logon	145
10.5.2	Users Being Added to the Wrong Backend Database	145
10.5.3	Configuration of auth methods	145
Chapter 11	GROUP MAPPING — MS WINDOWS AND UNIX	151
11.1	Features and Benefits	151
11.2	Discussion	153
11.2.1	Default Users, Groups and Relative Identifiers	154
11.2.2	Example Configuration	154
11.3	Configuration Scripts	155
11.3.1	Sample smb.conf Add Group Script	155
11.3.2	Script to Configure Group Mapping	156
11.4	Common Errors	156
11.4.1	Adding Groups Fails	157
11.4.2	Adding MS Windows Groups to MS Windows Groups Fails	157
11.4.3	Adding <i>Domain Users</i> to the <i>Power Users</i> Group	157
Chapter 12	FILE, DIRECTORY AND SHARE ACCESS CONTROLS	159
12.1	Features and Benefits	160
12.2	File System Access Controls	160
12.2.1	MS Windows NTFS Comparison with UNIX File Systems	160
12.2.2	Managing Directories	162
12.2.3	File and Directory Access Control	162
12.3	Share Definition Access Controls	164
12.3.1	User and Group-Based Controls	164
12.3.2	File and Directory Permissions-Based Controls	165
12.3.3	Miscellaneous Controls	165
12.4	Access Controls on Shares	165
12.4.1	Share Permissions Management	167
12.4.1.1	Windows NT4 Workstation/Server	167
12.4.1.2	Windows 200x/XP	168
12.5	MS Windows Access Control Lists and UNIX Interoperability	169
12.5.1	Managing UNIX Permissions Using NT Security Dialogs	169
12.5.2	Viewing File Security on a Samba Share	169
12.5.3	Viewing File Ownership	169
12.5.4	Viewing File or Directory Permissions	170
12.5.4.1	File Permissions	170
12.5.4.2	Directory Permissions	171
12.5.5	Modifying File or Directory Permissions	171
12.5.6	Interaction with the Standard Samba “ <i>create mask</i> ” Parameters	172
12.5.7	Interaction with the Standard Samba File Attribute Mapping	173
12.6	Common Errors	173
12.6.1	Users Cannot Write to a Public Share	173
12.6.2	File Operations Done as <i>root</i> with <i>force user</i> Set	175
12.6.3	MS Word with Samba Changes Owner of File	175

Chapter 13 FILE AND RECORD LOCKING	177
13.1 Features and Benefits	177
13.2 Discussion	177
13.2.1 Opportunistic Locking Overview	178
13.2.1.1 Exclusively Accessed Shares	180
13.2.1.2 Multiple-Accessed Shares or Files	180
13.2.1.3 UNIX or NFS Client-Accessed Files	181
13.2.1.4 Slow and/or Unreliable Networks	181
13.2.1.5 Multi-User Databases	181
13.2.1.6 PDM Data Shares	181
13.2.1.7 Beware of Force User	182
13.2.1.8 Advanced Samba Opportunistic Locking Parameters	182
13.2.1.9 Mission-Critical High-Availability	182
13.3 Samba Opportunistic Locking Control	183
13.3.1 Example Configuration	184
13.3.1.1 Disabling Oplocks	184
13.3.1.2 Disabling Kernel Oplocks	184
13.4 MS Windows Opportunistic Locking and Caching Controls	185
13.4.1 Workstation Service Entries	188
13.4.2 Server Service Entries	188
13.5 Persistent Data Corruption	189
13.6 Common Errors	189
13.6.1 locking.tdb Error Messages	189
13.6.2 Problems Saving Files in MS Office on Windows XP	190
13.6.3 Long Delays Deleting Files Over Network with XP SP1	190
13.7 Additional Reading	190
Chapter 14 SECURING SAMBA	191
14.1 Introduction	191
14.2 Features and Benefits	191
14.3 Technical Discussion of Protective Measures and Issues	192
14.3.1 Using Host-Based Protection	192
14.3.2 User-Based Protection	192
14.3.3 Using Interface Protection	192
14.3.4 Using a Firewall	193
14.3.5 Using IPC\$ Share-Based Denials	193
14.3.6 NTLMv2 Security	193
14.4 Upgrading Samba	194
14.5 Common Errors	194
14.5.1 Smbclient Works on Localhost, but the Network Is Dead	194
14.5.2 Why Can Users Access Home Directories of Other Users?	194
Chapter 15 INTERDOMAIN TRUST RELATIONSHIPS	197
15.1 Features and Benefits	197
15.2 Trust Relationship Background	197
15.3 Native MS Windows NT4 Trusts Configuration	198
15.3.1 Creating an NT4 Domain Trust	198
15.3.2 Completing an NT4 Domain Trust	198

15.3.3	Inter-Domain Trust Facilities	199
15.4	Configuring Samba NT-Style Domain Trusts	200
15.4.1	Samba as the Trusted Domain	200
15.4.2	Samba as the Trusting Domain	201
15.5	NT4-Style Domain Trusts with Windows 2000	201
15.6	Common Errors	202
15.6.1	Browsing of Trusted Domain Fails	202
Chapter 16	HOSTING A MICROSOFT DISTRIBUTED FILE SYSTEM TREE	203
16.1	Features and Benefits	203
16.2	Common Errors	204
16.2.1	MSDFS UNIX Path Is Case-Critical	204
Chapter 17	CLASSICAL PRINTING SUPPORT	207
17.1	Features and Benefits	207
17.2	Technical Introduction	208
17.2.1	Client to Samba Print Job Processing	208
17.2.2	Printing Related Configuration Parameters	209
17.3	Simple Print Configuration	209
17.3.1	Verifying Configuration with testparm	210
17.3.2	Rapid Configuration Validation	211
17.4	Extended Printing Configuration	213
17.4.1	Detailed Explanation Settings	214
17.4.1.1	The [global] Section	214
17.4.1.2	The [printers] Section	216
17.4.1.3	Any [my_printer_name] Section	217
17.4.1.4	Print Commands	218
17.4.1.5	Default UNIX System Printing Commands	218
17.4.1.6	Custom Print Commands	219
17.5	Printing Developments Since Samba-2.2	220
17.5.1	Point'n'Print Client Drivers on Samba Servers	221
17.5.2	The Obsolete [printer\$] Section	222
17.5.3	Creating the [print\$] Share	222
17.5.4	[print\$] Section Parameters	222
17.5.5	The [print\$] Share Directory	224
17.6	Installing Drivers into [print\$]	225
17.6.1	Add Printer Wizard Driver Installation	225
17.6.2	Installing Print Drivers Using rpcclient	226
17.6.2.1	Identifying Driver Files	226
17.6.2.2	Obtaining Driver Files from Windows Client [print\$] Shares	228
17.6.2.3	Installing Driver Files into [print\$]	229
17.6.2.4	smbclient to Confirm Driver Installation	230
17.6.2.5	Running rpcclient with adddriver	231
17.6.2.6	Checking adddriver Completion	232
17.6.2.7	Check Samba for Driver Recognition	233
17.6.2.8	Specific Driver Name Flexibility	234
17.6.2.9	Running rpcclient with the setdriver	234

17.7	Client Driver Installation Procedure	235
17.7.1	First Client Driver Installation	235
17.7.2	Setting Device Modes on New Printers	236
17.7.3	Additional Client Driver Installation	237
17.7.4	Always Make First Client Connection as root or “ <i>printer admin</i> ”	238
17.8	Other Gotchas	238
17.8.1	Setting Default Print Options for Client Drivers	239
17.8.2	Supporting Large Numbers of Printers	240
17.8.3	Adding New Printers with the Windows NT APW	242
17.8.4	Error Message: “ <i>Cannot connect under a different Name</i> ”	243
17.8.5	Take Care When Assembling Driver Files	243
17.8.6	Samba and Printer Ports	246
17.8.7	Avoiding Common Client Driver Misconfiguration	247
17.9	The Imprints Toolset	247
17.9.1	What is Imprints?	247
17.9.2	Creating Printer Driver Packages	247
17.9.3	The Imprints Server	247
17.9.4	The Installation Client	248
17.10	Adding Network Printers without User Interaction	248
17.11	The addprinter Command	250
17.12	Migration of Classical Printing to Samba	250
17.13	Publishing Printer Information in Active Directory or LDAP	251
17.14	Common Errors	251
17.14.1	I Give My Root Password but I Do Not Get Access	251
17.14.2	My Print Jobs Get Spooled into the Spooling Directory, but Then Get Lost	252
Chapter 18	CUPS PRINTING SUPPORT	253
18.1	Introduction	253
18.1.1	Features and Benefits	253
18.1.2	Overview	253
18.2	Basic CUPS Support Configuration	254
18.2.1	Linking <code>smbd</code> with <code>libcups.so</code>	254
18.2.2	Simple <code>smb.conf</code> Settings for CUPS	255
18.2.3	More Complex CUPS <code>smb.conf</code> Settings	256
18.3	Advanced Configuration	257
18.3.1	Central Spooling vs. “ <i>Peer-to-Peer</i> ” Printing	257
18.3.2	Raw Print Serving — Vendor Drivers on Windows Clients	257
18.3.3	Installation of Windows Client Drivers	257
18.3.4	Explicitly Enable “ <i>raw</i> ” Printing for <i>application/octet-stream</i>	258
18.3.5	Driver Upload Methods	259
18.4	Advanced Intelligent Printing with PostScript Driver Download	259
18.4.1	GDI on Windows – PostScript on UNIX	260
18.4.2	Windows Drivers, GDI and EMF	260
18.4.3	UNIX Printfile Conversion and GUI Basics	260
18.4.4	PostScript and Ghostscript	261
18.4.5	Ghostscript — the Software RIP for Non-PostScript Printers	262
18.4.6	PostScript Printer Description (PPD) Specification	263

18.4.7	Using Windows-Formatted Vendor PPDs	264
18.4.8	CUPS Also Uses PPDs for Non-PostScript Printers	265
18.5	The CUPS Filtering Architecture	265
18.5.1	MIME Types and CUPS Filters	266
18.5.2	MIME Type Conversion Rules	267
18.5.3	Filtering Overview	268
18.5.3.1	Filter requirements	268
18.5.4	Prefilters	268
18.5.5	pstops	269
18.5.6	pstoraster	270
18.5.7	imagetops and imagetoraster	270
18.5.8	rasterto [printers specific]	271
18.5.9	CUPS Backends	271
18.5.10	The Role of cupsomatic/foomatic	274
18.5.11	The Complete Picture	275
18.5.12	mime.convs	275
18.5.13	“Raw” Printing	275
18.5.14	application/octet-stream Printing	276
18.5.15	PostScript Printer Descriptions (PPDs) for Non-PS Printers	277
18.5.16	<i>cupsomatic/foomatic-rip</i> Versus <i>native CUPS</i> Printing	277
18.5.17	Examples for Filtering Chains	279
18.5.18	Sources of CUPS Drivers/PPDs	280
18.5.19	Printing with Interface Scripts	281
18.6	Network Printing (Purely Windows)	281
18.6.1	From Windows Clients to an NT Print Server	281
18.6.2	Driver Execution on the Client	282
18.6.3	Driver Execution on the Server	282
18.7	Network Printing (Windows Clients — UNIX/Samba Print Servers)	282
18.7.1	From Windows Clients to a CUPS/Samba Print Server	283
18.7.2	Samba Receiving Jobfiles and Passing Them to CUPS	283
18.8	Network PostScript RIP	284
18.8.1	PPDs for Non-PS Printers on UNIX	285
18.8.2	PPDs for Non-PS Printers on Windows	285
18.9	Windows Terminal Servers (WTS) as CUPS Clients	285
18.9.1	Printer Drivers Running in “ <i>Kernel Mode</i> ” Cause Many Problems	285
18.9.2	Workarounds Impose Heavy Limitations	286
18.9.3	CUPS: A “ <i>Magical Stone</i> ”?	286
18.9.4	PostScript Drivers with No Major Problems — Even in Kernel Mode	286
18.10	Configuring CUPS for Driver Download	287
18.10.1	<i>cupsaddsbm</i> : The Unknown Utility	287
18.10.2	Prepare Your smb.conf for cupsaddsbm	287
18.10.3	CUPS “ <i>PostScript Driver for Windows NT/200x/XP</i> ”	287
18.10.4	Recognizing Different Driver Files	289
18.10.5	Acquiring the Adobe Driver Files	290
18.10.6	ESP Print Pro PostScript Driver for Windows NT/200x/XP	290
18.10.7	Caveats to be Considered	290
18.10.8	Windows CUPS PostScript Driver Versus Adobe Driver	292
18.10.9	Run cupsaddsbm (Quiet Mode)	293

18.10.10	Run cupsaddsmb with Verbose Output	293
18.10.11	Understanding cupsaddsmb	295
18.10.12	How to Recognize If cupsaddsmb Completed Successfully	296
18.10.13	cupsaddsmb with a Samba PDC	296
18.10.14	cupsaddsmb Flowchart	297
18.10.15	Installing the PostScript Driver on a Client	297
18.10.16	Avoiding Critical PostScript Driver Settings on the Client	298
18.11	Installing PostScript Driver Files Manually Using rpcclient	299
18.11.1	A Check of the rpcclient man Page	299
18.11.2	Understanding the rpcclient man Page	300
18.11.3	Producing an Example by Querying a Windows Box	300
18.11.4	Requirements for adddriver and setdriver to Succeed	301
18.11.5	Manual Driver Installation in 15 Steps	302
18.11.6	Troubleshooting Revisited	307
18.12	The Printing *.tdb Files	308
18.12.1	Trivial Database Files	308
18.12.2	Binary Format	308
18.12.3	Losing *.tdb Files	308
18.12.4	Using tdbbackup	309
18.13	CUPS Print Drivers from Linuxprinting.org	309
18.13.1	foomatic-rip and Foomatic Explained	310
18.13.1.1	690 “ <i>Perfect</i> ” Printers	310
18.13.1.2	How the Printing HOWTO Started It All	311
18.13.1.3	Foomatic’s Strange Name	311
18.13.1.4	cupsomatic, pdqomatic, lpdomatic, directomatic	311
18.13.1.5	The <i>Grand Unification</i> Achieved	312
18.13.1.6	Driver Development Outside	313
18.13.1.7	Forums, Downloads, Tutorials, Howtos — also for Mac OS X and Commercial UNIX	313
18.13.1.8	Foomatic Database-Generated PPDs	314
18.13.2	foomatic-rip and Foomatic-PPD Download and Installation	314
18.14	Page Accounting with CUPS	317
18.14.1	Setting Up Quotas	317
18.14.2	Correct and Incorrect Accounting	317
18.14.3	Adobe and CUPS PostScript Drivers for Windows Clients	317
18.14.4	The page_log File Syntax	318
18.14.5	Possible Shortcomings	319
18.14.6	Future Developments	319
18.15	Additional Material	319
18.16	Auto-Deletion or Preservation of CUPS Spool Files	321
18.16.1	CUPS Configuration Settings Explained	321
18.16.2	Pre-Conditions	321
18.16.3	Manual Configuration	322
18.17	Printing from CUPS to Windows Attached Printers	322
18.18	More CUPS-Filtering Chains	323
18.19	Common Errors	324
18.19.1	Windows 9x/ME Client Can’t Install Driver	324
18.19.2	“ <i>cupsaddsmb</i> ” Keeps Asking for Root Password in Never-ending Loop	324

18.19.3 “ <i>cupsaddsmb</i> ” Errors	324
18.19.4 Client Can’t Connect to Samba Printer	325
18.19.5 New Account Reconnection from Windows 200x/XP Troubles	325
18.19.6 Avoid Being Connected to the Samba Server as the Wrong User	325
18.19.7 Upgrading to CUPS Drivers from Adobe Drivers	325
18.19.8 Can’t Use “ <i>cupsaddsmb</i> ” on Samba Server Which Is a PDC	326
18.19.9 Deleted Windows 200x Printer Driver Is Still Shown	326
18.19.10 Windows 200x/XP ”Local Security Policies”	326
18.19.11 Administrator Cannot Install Printers for All Local Users	326
18.19.12 Print Change Notify Functions on NT-clients	326
18.19.13 WinXP-SP1	326
18.19.14 Print Options for All Users Can’t Be Set on Windows 200x/XP	326
18.19.15 Most Common Blunders in Driver Settings on Windows Clients	327
18.19.16 cupsaddsmb Does Not Work with Newly Installed Printer	328
18.19.17 Permissions on /var/spool/samba/ Get Reset After Each Reboot	328
18.19.18 Print Queue Called “ <i>lp</i> ” Mis-handles Print Jobs	328
18.19.19 Location of Adobe PostScript Driver Files for “ <i>cupsaddsmb</i> ”	328
18.20 Overview of the CUPS Printing Processes	328
Chapter 19 STACKABLE VFS MODULES	331
19.1 Features and Benefits	331
19.2 Discussion	331
19.3 Included Modules	332
19.3.1 audit	332
19.3.2 extd_audit	332
19.3.3 fake_perms	332
19.3.4 recycle	333
19.3.5 netatalk	333
19.4 VFS Modules Available Elsewhere	334
19.4.1 DatabaseFS	334
19.4.2 vscan	334
Chapter 20 WINBIND: USE OF DOMAIN ACCOUNTS	335
20.1 Features and Benefits	335
20.2 Introduction	336
20.3 What Winbind Provides	336
20.3.1 Target Uses	337
20.4 How Winbind Works	337
20.4.1 Microsoft Remote Procedure Calls	337
20.4.2 Microsoft Active Directory Services	338
20.4.3 Name Service Switch	338
20.4.4 Pluggable Authentication Modules	339
20.4.5 User and Group ID Allocation	339
20.4.6 Result Caching	339
20.5 Installation and Configuration	340
20.5.1 Introduction	340
20.5.2 Requirements	340
20.5.3 Testing Things Out	341

20.5.3.1	Configure nsswitch.conf and the Winbind Libraries on Linux and Solaris	341
20.5.3.2	NSS Winbind on AIX	342
20.5.3.3	Configure smb.conf	342
20.5.3.4	Join the Samba Server to the PDC Domain	342
20.5.3.5	Starting and Testing the winbindd Daemon	342
20.5.3.6	Fix the init.d Startup Scripts	344
20.5.3.7	Configure Winbind and PAM	347
20.6	Conclusion	350
20.7	Common Errors	351
20.7.1	NSCD Problem Warning	351
20.7.2	Winbind Is Not Resolving Users and Groups	351
Chapter 21	ADVANCED NETWORK MANAGEMENT	353
21.1	Features and Benefits	353
21.2	Remote Server Administration	353
21.3	Remote Desktop Management	354
21.3.1	Remote Management from NoMachine.Com	354
21.4	Network Logon Script Magic	355
21.4.1	Adding Printers without User Intervention	357
Chapter 22	SYSTEM AND ACCOUNT POLICIES	359
22.1	Features and Benefits	359
22.2	Creating and Managing System Policies	359
22.2.1	Windows 9x/ME Policies	360
22.2.2	Windows NT4-Style Policy Files	361
22.2.2.1	Registry Spoiling	361
22.2.3	MS Windows 200x/XP Professional Policies	361
22.2.3.1	Administration of Windows 200x/XP Policies	362
22.3	Managing Account/User Policies	363
22.4	Management Tools	364
22.4.1	Samba Editreg Toolset	364
22.4.2	Windows NT4/200x	364
22.4.3	Samba PDC	364
22.5	System Startup and Logon Processing Overview	364
22.6	Common Errors	365
22.6.1	Policy Does Not Work	365
Chapter 23	DESKTOP PROFILE MANAGEMENT	367
23.1	Features and Benefits	367
23.2	Roaming Profiles	367
23.2.1	Samba Configuration for Profile Handling	368
23.2.1.1	NT4/200x User Profiles	368
23.2.1.2	Windows 9x/Me User Profiles	368
23.2.1.3	Mixed Windows 9x/Me and Windows NT4/200x User Profiles	369
23.2.1.4	Disabling Roaming Profile Support	369
23.2.2	Windows Client Profile Configuration Information	370
23.2.2.1	Windows 9x/Me Profile Setup	370

23.2.2.2	Windows NT4 Workstation	372
23.2.2.3	Windows 2000/XP Professional	372
23.2.3	Sharing Profiles between W9x/Me and NT4/200x/XP Workstations	374
23.2.4	Profile Migration from Windows NT4/200x Server to Samba	374
23.2.4.1	Windows NT4 Profile Management Tools	374
23.2.4.2	Side Bar Notes	375
23.2.4.3	moveuser.exe	375
23.2.4.4	Get SID	375
23.3	Mandatory Profiles	376
23.4	Creating and Managing Group Profiles	376
23.5	Default Profile for Windows Users	377
23.5.1	MS Windows 9x/Me	377
23.5.1.1	User Profile Handling with Windows 9x/Me	377
23.5.2	MS Windows NT4 Workstation	377
23.5.3	MS Windows 200x/XP	380
23.6	Common Errors	382
23.6.1	Configuring Roaming Profiles for a Few Users or Groups	382
23.6.2	Cannot Use Roaming Profiles	382
23.6.3	Changing the Default Profile	384
Chapter 24	PAM-BASED DISTRIBUTED AUTHENTICATION	385
24.1	Features and Benefits	385
24.2	Technical Discussion	387
24.2.1	PAM Configuration Syntax	387
24.2.1.1	Anatomy of /etc/pam.d Entries	387
24.2.2	Example System Configurations	391
24.2.2.1	PAM: Original Login Config	391
24.2.2.2	PAM: Login Using pam_smbpass	392
24.2.3	smb.conf PAM Configuration	393
24.2.4	Remote CIFS Authentication Using winbindd.so	394
24.2.5	Password Synchronization Using pam_smbpass.so	394
24.2.5.1	Password Synchronization Configuration	395
24.2.5.2	Password Migration Configuration	396
24.2.5.3	Mature Password Configuration	396
24.2.5.4	Kerberos Password Integration Configuration	396
24.3	Common Errors	397
24.3.1	pam_winbind Problem	397
24.3.2	Winbind Is Not Resolving Users and Groups	397
Chapter 25	INTEGRATING MS WINDOWS NETWORKS WITH SAMBA	399
25.1	Features and Benefits	399
25.2	Background Information	399
25.3	Name Resolution in a Pure UNIX/Linux World	400
25.3.1	/etc/hosts	400
25.3.2	/etc/resolv.conf	401
25.3.3	/etc/host.conf	401
25.3.4	/etc/nsswitch.conf	402
25.4	Name Resolution as Used within MS Windows Networking	403

25.4.1	The NetBIOS Name Cache	404
25.4.2	The LMHOSTS File	404
25.4.3	HOSTS File	406
25.4.4	DNS Lookup	406
25.4.5	WINS Lookup	406
25.5	Common Errors	407
25.5.1	Pinging Works Only in One Way	407
25.5.2	Very Slow Network Connections	407
25.5.3	Samba Server Name Change Problem	408
Chapter 26	UNICODE/CHARSETS	409
26.1	Features and Benefits	409
26.2	What Are Charsets and Unicode?	409
26.3	Samba and Charsets	410
26.4	Conversion from Old Names	410
26.5	Common Errors	410
26.5.1	CP850.so Can't Be Found	410
Chapter 27	BACKUP TECHNIQUES	413
27.1	Features and Benefits	413
27.2	Discussion of Backup Solutions	413
27.2.1	BackupPC	413
27.2.2	Rsync	414
27.2.3	Amanda	414
27.2.4	BOBS: Browseable Online Backup System	415
Chapter 28	HIGH AVAILABILITY	417
28.1	Features and Benefits	417
28.2	Technical Discussion	417
28.2.1	The Ultimate Goal	417
28.2.2	Why Is This So Hard?	418
28.2.2.1	The Front-End Challenge	418
28.2.2.2	De-multiplexing SMB Requests	419
28.2.2.3	The Distributed File System Challenge	419
28.2.2.4	Restrictive Constraints on Distributed File Systems	419
28.2.2.5	Server Pool Communications	420
28.2.2.6	Server Pool Communications Demands	420
28.2.2.7	Required Modifications to Samba	420
28.2.3	A Simple Solution	420
28.2.4	High Availability Server Products	421
28.2.5	MS-DFS: The Poor Man's Cluster	421
28.2.6	Conclusions	421
Part IV	Migration and Updating	421
Chapter 29	UPGRADING FROM SAMBA-2.X TO SAMBA-3.0.0	423
29.1	Quick Migration Guide	423
29.2	New Features in Samba-3	423

29.3	Configuration Parameter Changes	424
29.3.1	Removed Parameters	424
29.3.2	New Parameters	425
29.3.3	Modified Parameters (Changes in Behavior):	427
29.4	New Functionality	428
29.4.1	Databases	428
29.4.2	Changes in Behavior	428
29.4.3	Passdb Backends and Authentication	429
29.4.4	LDAP	429
29.4.4.1	New Schema	429
29.4.4.2	New Suffix for Searching	430
29.4.4.3	IdMap LDAP Support	430
Chapter 30	MIGRATION FROM NT4 PDC TO SAMBA-3 PDC	433
30.1	Planning and Getting Started	433
30.1.1	Objectives	433
30.1.1.1	Domain Layout	434
30.1.1.2	Server Share and Directory Layout	435
30.1.1.3	Logon Scripts	436
30.1.1.4	Profile Migration/Creation	436
30.1.1.5	User and Group Accounts	436
30.1.2	Steps in Migration Process	436
30.2	Migration Options	437
30.2.1	Planning for Success	437
30.2.2	Samba-3 Implementation Choices	438
Chapter 31	SWAT — THE SAMBA WEB ADMINISTRATION TOOL	441
31.1	Features and Benefits	441
31.2	Guidelines and Technical Tips	442
31.2.1	Validate SWAT Installation	442
31.2.1.1	Locating the swat File	442
31.2.1.2	Locating the SWAT Support Files	443
31.2.2	Enabling SWAT for Use	444
31.2.3	Securing SWAT through SSL	445
31.2.4	Enabling SWAT Internationalization Support	446
31.3	Overview and Quick Tour	447
31.3.1	The SWAT Home Page	447
31.3.2	Global Settings	447
31.3.3	Share Settings	448
31.3.4	Printers Settings	448
31.3.5	The SWAT Wizard	448
31.3.6	The Status Page	449
31.3.7	The View Page	449
31.3.8	The Password Change Page	449
31.4	SWAT View Page Displays Incorrectly	449

Part V Troubleshooting	449
Chapter 32 THE SAMBA CHECKLIST	451
32.1 Introduction	451
32.2 Assumptions	451
32.3 The Tests	452
Chapter 33 ANALYZING AND SOLVING SAMBA PROBLEMS	459
33.1 Diagnostics Tools	459
33.1.1 Debugging with Samba Itself	459
33.1.2 Tcpdump	460
33.1.3 Ethereal	460
33.1.4 The Windows Network Monitor	461
33.1.4.1 Installing Network Monitor on an NT Workstation	461
33.1.4.2 Installing Network Monitor on Windows 9x/Me	462
33.2 Useful URLs	462
33.3 Getting Mailing List Help	462
33.4 How to Get Off the Mailing Lists	463
Chapter 34 REPORTING BUGS	465
34.1 Introduction	465
34.2 General Information	465
34.3 Debug Levels	465
34.4 Internal Errors	466
34.5 Attaching to a Running Process	467
34.6 Patches	467
Part VI Appendixes	467
Chapter 35 HOW TO COMPILE SAMBA	469
35.1 Access Samba Source Code via CVS	469
35.1.1 Introduction	469
35.1.2 CVS Access to samba.org	469
35.1.2.1 Access via CVSweb	469
35.1.2.2 Access via CVS	470
35.2 Accessing the Samba Sources via rsync and ftp	470
35.3 Verifying Samba's PGP Signature	470
35.4 Building the Binaries	471
35.4.1 Compiling Samba with Active Directory Support	472
35.4.1.1 Installing the Required Packages for Debian	472
35.4.1.2 Installing the Required Packages for Red Hat Linux	473
35.4.1.3 SuSE Linux Package Requirements	473
35.5 Starting the smbd and nmbd	473
35.5.1 Starting from inetd.conf	473
35.5.2 Alternative: Starting smbd as a Daemon	475
Chapter 36 PORTABILITY	477
36.1 HPUX	477

36.2	SCO UNIX	477
36.3	DNIX	478
36.4	Red Hat Linux	479
36.5	AIX	479
36.5.1	Sequential Read Ahead	479
36.6	Solaris	480
36.6.1	Locking Improvements	480
36.6.2	Winbind on Solaris 9	480
Chapter 37	SAMBA AND OTHER CIFS CLIENTS	481
37.1	Macintosh Clients	481
37.2	OS2 Client	481
37.2.1	Configuring OS/2 Warp Connect or OS/2 Warp 4	481
37.2.2	Configuring Other Versions of OS/2	482
37.2.3	Printer Driver Download for OS/2 Clients	482
37.3	Windows for Workgroups	483
37.3.1	Latest TCP/IP Stack from Microsoft	483
37.3.2	Delete .pwl Files After Password Change	483
37.3.3	Configuring Windows for Workgroups Password Handling	483
37.3.4	Password Case Sensitivity	483
37.3.5	Use TCP/IP as Default Protocol	483
37.3.6	Speed Improvement	484
37.4	Windows 95/98	484
37.4.1	Speed Improvement	484
37.5	Windows 2000 Service Pack 2	484
37.6	Windows NT 3.1	485
Chapter 38	SAMBA PERFORMANCE TUNING	487
38.1	Comparisons	487
38.2	Socket Options	487
38.3	Read Size	488
38.4	Max Xmit	488
38.5	Log Level	488
38.6	Read Raw	489
38.7	Write Raw	489
38.8	Slow Logins	489
38.9	Client Tuning	489
38.10	Samba Performance Problem Due to Changing Linux Kernel	489
38.11	Corrupt tdb Files	490
38.12	Samba Performance is Very Slow	490
Chapter 39	DNS AND DHCP CONFIGURATION GUIDE	491
39.1	Features and Benefits	491
39.2	Example Configuration	491
39.2.1	Dynamic DNS	492
39.2.2	DHCP Server	495
Chapter A	MANUAL PAGES	497
A.1	smb.conf	497

A.2 nmblookup	597
A.3 rpcclient	600
A.4 smbcacls	608
A.5 smbclient	612
A.6 net	623
A.7 nmbd	633
A.8 pdbedit	636
A.9 smbcquotas	641
A.10 smbdrive	644
A.11 smbpasswd	648
A.12 smbpasswd	651
A.13 smbstatus	655
A.14 smbtree	656
A.15 testparm	658
A.16 wbinfos	660
A.17 winbindd	662
Chapter B THE GNU GENERAL PUBLIC LICENSE	669
GLOSSARY	677
SUBJECT INDEX	681

List of Figures

4.1	An Example Domain.	41
8.1	Network Bridge Configuration.	86
8.2	Internet Protocol (TCP/IP) Properties.	87
8.3	Advanced Network Settings	88
8.4	DNS Configuration.	89
8.5	WINS Configuration	90
8.6	Local Area Connection Properties.	91
8.7	Internet Protocol (TCP/IP) Properties.	91
8.8	Advanced Network Settings.	92
8.9	DNS Configuration.	93
8.10	WINS Configuration.	94
8.11	The Windows Me Network Configuration Panel.	95
8.12	IP Address.	96
8.13	DNS Configuration.	96
8.14	WINS Configuration.	97
8.15	The General Panel.	97
8.16	The Computer Name Panel.	98
8.17	The Computer Name Changes Panel.	98
8.18	The Computer Name Changes Panel Domain MIDEARTH.	99
8.19	Computer Name Changes User name and Password Panel.	99
8.20	The Network Panel.	100
8.21	Client for Microsoft Networks Properties Panel.	100
8.22	Identification Panel.	101
8.23	Identification Panel.	101
9.1	Cross-Subnet Browsing Example.	119
10.1	IDMAP: Resolution of SIDs to UIDs.	127
10.2	IDMAP: Resolution of UIDs to SIDs.	128
11.1	IDMAP: group SID to GID resolution.	152
11.2	IDMAP: GID resolution to matching SID.	152
11.3	IDMAP storing group mappings.	152
12.1	Overview of UNIX permissions field.	163

15.1	Trusts overview.	199
18.1	Windows printing to a local printer.	261
18.2	Printing to a PostScript printer.	262
18.3	Ghostscript as a RIP for non-postscript printers.	263
18.4	Pre-filtering in CUPS to form PostScript.	269
18.5	Adding device-specific print options.	269
18.6	PostScript to intermediate raster format.	270
18.7	CUPS-raster production using Ghostscript.	271
18.8	Image format to CUPS-raster format conversion.	272
18.9	Raster to printer-specific formats.	273
18.10	cupsomatic/foomatic Processing versus Native CUPS.	278
18.11	PDF to socket chain.	279
18.12	PDF to USB chain.	280
18.13	Print driver execution on the client.	282
18.14	Print driver execution on the server.	283
18.15	Printing via CUPS/Samba server.	284
18.16	cupsaddsmb flowchart.	297
18.17	Filtering chain 1.	324
18.18	Filtering chain with cupsomatic	329
18.19	CUPS printing overview.	330
33.1	Starting a capture.	460
33.2	Main ethereal data window.	461

List of Tables

5.1	Domain Backend Account Distribution Options	58
6.1	Assumptions	72
9.1	Browse Subnet Example 1	120
9.2	Browse Subnet Example 2	120
9.3	Browse Subnet Example 3	121
9.4	Browse Subnet Example 4	121
10.1	Attributes in the sambaSamAccount objectclass (LDAP) — Part A	147
10.2	Attributes in the sambaSamAccount objectclass (LDAP) — Part B	148
10.3	Possible <i>ldap passwd sync</i> values	148
10.4	Basic smb.conf options for MySQL passdb backend	148
10.5	MySQL field names for MySQL passdb backend	149
11.1	Well-Known User Default RIDs	155
12.1	Managing Directories with UNIX and Windows	162
12.2	User and Group Based Controls	165
12.3	File and Directory Permission Based Controls	166
12.4	Other Controls	167
17.1	Default Printing Settings	219
18.1	PPDs shipped with CUPS	277
19.1	Extended Auditing Log Information	332
23.1	User Shell Folder Registry Keys Default Values	379
23.2	Defaults of Profile Settings Registry Keys	379
23.3	Defaults of Default User Profile Paths Registry Keys	381
24.1	Options recognized by pam_smbpass	395
25.1	Unique NetBIOS Names	403
25.2	Group Names	403

29.1 TDB File Descriptions	428
30.1 The Three Major Site Types	437
30.2 Nature of the Conversion Choices	438

Part I

General Installation

PREFACE AND INTRODUCTION

“A man’s gift makes room for him before great men. Gifts are like hooks that can catch hold of the mind taking it beyond the reach of forces that otherwise might constrain it.” — Anon.

This is a book about Samba. It is a tool, a derived work of the labors of many and of the diligence and goodwill of more than a few. This book contains material that has been contributed in a persistent belief that each of us can add value to our neighbors as well as to those who will follow us.

This book is designed to meet the needs of the Microsoft network administrator. UNIX administrators will benefit from this book also, though they may complain that it is hard to find the information they think they need. So if you are a Microsoft certified specialist, this book should meet your needs rather well. If you are a UNIX or Linux administrator, there is no need to feel badly — you should have no difficulty finding answers to your current concerns also.

What Is Samba?

Samba is a big, complex project. The Samba project is ambitious and exciting. The team behind Samba is a group of some thirty individuals who are spread the world over and come from an interesting range of backgrounds. This team includes scientists, engineers, programmers, business people, and students.

Team members were drawn into active participation through the desire to help deliver an exciting level of transparent interoperability between Microsoft Windows and the non-Microsoft information technology world.

The slogan that unites the efforts behind the Samba project says: *Samba, Opening Windows to a Wider World!* The goal behind the project is one of removing barriers to interoperability.

Samba provides file and print services for Microsoft Windows clients. These services may be hosted off any TCP/IP-enabled platform. The original deployment platforms were UNIX and Linux, though today it is in common use across a broad variety of systems.

The Samba project includes not only an impressive feature set in file and print serving capabilities, but has been extended to include client functionality, utilities to ease migration to Samba, tools to aid interoperability with Microsoft Windows, and administration tools.

The real people behind Samba are users like you. You have inspired the developers (the Samba Team) to do more than any of them imagined could or should be done. User feedback drives Samba development. Samba-3 in particular incorporates a huge amount of work done as a result of user requests, suggestions and direct code contributions.

Why This Book?

There is admittedly a large number of Samba books on the market today and each book has its place. Despite the apparent plethora of books, Samba as a project continues to receive much criticism for failing to provide sufficient documentation. Samba is also criticized for being too complex and too difficult to configure. In many ways this is evidence of the success of Samba as there would be no complaints if it was not successful.

The Samba Team members work predominantly with UNIX and Linux, so it is hardly surprising that existing Samba documentation should reflect that orientation. The original HOWTO text documents were intended to provide some tips, a few golden nuggets, and if they helped anyone then that was just wonderful. But the HOWTOs lacked structure and context. They were isolated snapshots of information that were written to pass information on to someone else who might benefit. They reflected a need to transmit more information that could be conveniently put into manual pages.

The original HOWTO documents were written by different authors. Most HOWTO documents are the result of feedback and contributions from numerous authors. In this book we took care to preserve as much original content as possible. As you read this book you will note that chapters were written by multiple authors, each of whom has his own style. This demonstrates the nature of the Open Source software development process.

Out of the original HOWTO documents sprang a collection of unofficial HOWTO documents that are spread over the Internet. It is sincerely intended that this work will *not* replace the valuable unofficial HOWTO work that continues to flourish. If you are involved in unofficial HOWTOs then please continue your work!

Those of you who have dedicated your labors to the production of unofficial HOWTOs, to Web page information regarding Samba, or to answering questions on the mailing lists or elsewhere, may be aware that this is a labor of love. We would like to know about your contribution and willingly receive the precious pearls of wisdom you have collected. Please email your contribution to John H. Terpstra (jht@samba.org). As a service to other users we will gladly adopt material that is technically accurate.

Existing Samba books are largely addressed to the UNIX administrator. From the perspective of this target group the existing books serve an adequate purpose, with one exception — now that Samba-3 is out they need to be updated!

This book, the *Official Samba-3 HOWTO and Reference Guide*, includes the Samba-HOWTO-Collection.pdf that ships with Samba. These documents have been written with a new design intent and purpose.

Over the past two years many Microsoft network administrators have adopted Samba and have become interested in its deployment. Their information needs are very different from that of the UNIX administrator. This book has been arranged and the information presented

from the perspective of someone with previous Microsoft Windows network administrative training and experience.

Book Structure and Layout

This book is presented in six parts:

General Installation — Designed to help you get Samba-3 running quickly. The Fast Start chapter is a direct response to requests from Microsoft network administrators for some sample configurations that *just work*.

Server Configuration Basics — The purpose of this section is to aid the transition from existing Microsoft Windows network knowledge to Samba terminology and norms. The chapters in this part each cover the installation of one type of Samba server.

Advanced Configuration — The mechanics of network browsing have long been the Achilles heel of all Microsoft Windows users. Samba-3 introduces new user and machine account management facilities, a new way to map UNIX groups and Windows groups, Interdomain trusts, new loadable file system drivers (VFS), and more. New with this document is expanded printing documentation, as well as a wealth of information regarding desktop and user policy handling, use of desktop profiles, and techniques for enhanced network integration. This section makes up the core of the book. Read and enjoy.

Migration and Updating — A much requested addition to the book is information on how to migrate from Microsoft Windows NT4 to Samba-3, as well as an overview of what the issues are when moving from Samba-2.x to Samba-3.

Troubleshooting — This short section should help you when all else fails.

Appendix — Here you will find a collection of things that are either too peripheral for most users, or are a little left of field to be included in the main body of information.

Welcome to Samba-3 and the first published document to help you and your users to enjoy a whole new world of interoperability between Microsoft Windows and the rest of the world.

HOW TO INSTALL AND TEST SAMBA

1.1 Obtaining and Installing Samba

Binary packages of Samba are included in almost any Linux or UNIX distribution. There are also some packages available at the Samba homepage¹. Refer to the manual of your operating system for details on installing packages for your specific operating system.

If you need to compile Samba from source, check Chapter 35, *How to Compile Samba*.

1.2 Configuring Samba (smb.conf)

Samba's configuration is stored in the `smb.conf` file, which usually resides in `/etc/samba/smb.conf` or `/usr/local/samba/lib/smb.conf`. You can either edit this file yourself or do it using one of the many graphical tools that are available, such as the Web-based interface SWAT, that is included with Samba.

1.2.1 Configuration file syntax

The `smb.conf` file uses the same syntax as the various old `.ini` files in Windows 3.1: Each file consists of various sections, which are started by putting the section name between brackets (`[]`) on a new line. Each contains zero or more key/value-pairs separated by an equality sign (`=`). The file is just a plain-text file, so you can open and edit it with your favorite editing tool.

Each section in the `smb.conf` file represents a share on the Samba server. The section "*global*" is special, since it contains settings that apply to the whole Samba server and not to one share in particular.

Example 1.1 contains a very minimal `smb.conf`.

¹<http://samba.org/>

Example 1.1. A minimal smb.conf

```
[global]
workgroup = WKG
netbios name = MYNAME

[share1]
path = /tmp

[share2]
path = /my_shared_folder
comment = Some random files
```

1.2.2 Example Configuration

There are sample configuration files in the examples subdirectory in the distribution. It is suggested you read them carefully so you can see how the options go together in practice. See the man page for all the options. It might be worthwhile to start out with the smb.conf.default configuration file and adapt it to your needs. It contains plenty of comments.

The simplest useful configuration file would contain something like shown in Example 1.2.

Example 1.2. Another simple smb.conf File

```
[global]
workgroup = MIDEARTH

[homes]
guest ok = no
read only = no
```

This will allow connections by anyone with an account on the server, using either their login name or *homes* as the service name. (Note: The workgroup that Samba should appear in must also be set. The default workgroup name is WORKGROUP.)

Make sure you put the `smb.conf` file in the correct place.

For more information about security settings for the *[homes]* share please refer to Chapter 14, *Securing Samba*.

1.2.2.1 Test Your Config File with testparm

It's important to validate the contents of the `smb.conf` file using the `testparm` program. If `testparm` runs correctly, it will list the loaded services. If not, it will give an error message. Make sure it runs correctly and that the services look reasonable before proceeding. Enter the command:

```
root# testparm /etc/samba/smb.conf
```

Testparm will parse your configuration file and report any unknown parameters or incorrect syntax.

Always run testparm again whenever the `smb.conf` file is changed!

1.2.3 SWAT

SWAT is a Web-based interface that can be used to facilitate the configuration of Samba. SWAT might not be available in the Samba package that shipped with your platform, but in a separate package. Please read the SWAT manpage on compiling, installing and configuring SWAT from source.

To launch SWAT, just run your favorite Web browser and point it to `http://localhost:901/`. Replace *localhost* with the name of the computer on which Samba is running if that is a different computer than your browser.

SWAT can be used from a browser on any IP-connected machine, but be aware that connecting from a remote machine leaves your connection open to password sniffing as passwords will be sent over the wire in the clear.

More information about SWAT can be found in Chapter 31, *SWAT The Samba Web Administration Tool*.

1.3 List Shares Available on the Server

To list shares that are available from the configured Samba server execute the following command:

```
$ smbclient -L yourhostname
```

You should see a list of shares available on your server. If you do not, then something is incorrectly configured. This method can also be used to see what shares are available on other SMB servers, such as Windows 2000.

If you choose user-level security you may find that Samba requests a password before it will list the shares. See the `smbclient` man page for details. You can force it to list the shares without a password by adding the option `-N` to the command line.

1.4 Connect with a UNIX Client

Enter the following command:

```
$ smbclient //yourhostname/aservice
```

Typically *yourhostname* is the name of the host on which `smbd` has been installed. The *aservice* is any service that has been defined in the `smb.conf` file. Try your user name if you just have a `[homes]` section in the `smb.conf` file.

Example: If the UNIX host is called *bambi* and a valid login name is *fred*, you would type:

```
$ smbclient //bambi/fred
```

1.5 Connect from a Remote SMB Client

Now that Samba is working correctly locally, you can try to access it from other clients. Within a few minutes, the Samba host should be listed in the Network Neighborhood on all Windows clients of its subnet. Try browsing the server from another client or 'mounting' it.

Mounting disks from a DOS, Windows or OS/2 client can be done by running a command such as:

```
C:\> net use d: \\servername\service
```

Try printing, e.g.

```
C:\> net use lpt1: \\servername\spoolservice
```

```
C:\> print filename
```

1.6 What If Things Don't Work?

You might want to read Chapter 32, *The Samba Checklist*. If you are still stuck, refer to Chapter 33, *Analyzing and Solving Samba Problems*. Samba has been successfully installed at thousands of sites worldwide. It is unlikely that your particular problem is unique, so it might be productive to perform an Internet search to see if someone else has encountered your problem and has found a way to overcome it.

1.7 Common Errors

The following questions and issues are raised repeatedly on the Samba mailing list.

1.7.1 Large Number of `smbd` Processes

Samba consists of three core programs: `nmbd`, `smbd`, and `winbindd`. `nmbd` is the name server message daemon, `smbd` is the server message daemon, and `winbindd` is the daemon that handles communication with Domain Controllers.

If Samba is *not* running as a WINS server, then there will be one single instance of `nmbd` running on your system. If it is running as a WINS server then there will be two instances — one to handle the WINS requests.

`smbd` handles all connection requests. It spawns a new process for each client connection made. That is why you may see so many of them, one per client connection.

`winbindd` will run as one or two daemons, depending on whether or not it is being run in *split mode* (in which case there will be two instances).

1.7.2 Error Message: `open_oplock_ipc`

An error message is observed in the log files when `smbd` is started: “*open_oplock_ipc: Failed to get local UDP socket for address 100007f. Error was Cannot assign requested.*”

Your loopback device isn’t working correctly. Make sure it is configured correctly. The loopback device is an internal (virtual) network device with the IP address `127.0.0.1`. Read your OS documentation for details on how to configure the loopback on your system.

1.7.3 “The network name cannot be found”

This error can be caused by one of these misconfigurations:

- You specified a nonexistent path for the share in `smb.conf`.
- The user you are trying to access the share with does not have sufficient permissions to access the path for the share. Both read (r) and access (x) should be possible.
- The share you are trying to access does not exist.

FAST START: CURE FOR IMPATIENCE

When we first asked for suggestions for inclusion in the Samba HOWTO documentation, someone wrote asking for example configurations — and lots of them. That is remarkably difficult to do, without losing a lot of value that can be derived from presenting many extracts from working systems. That is what the rest of this document does. It does so with extensive descriptions of the configuration possibilities within the context of the chapter that covers it. We hope that this chapter is the medicine that has been requested.

2.1 Features and Benefits

Samba needs very little configuration to create a basic working system. In this chapter we progress from the simple to the complex, for each providing all steps and configuration file changes needed to make each work. Please note that a comprehensively configured system will likely employ additional smart features. The additional features are covered in the remainder of this document.

The examples used here have been obtained from a number of people who made requests for example configurations. All identities have been obscured to protect the guilty and any resemblance to unreal non-existent sites is deliberate.

2.2 Description of Example Sites

In the first set of configuration examples we consider the case of exceptionally simple system requirements. There is a real temptation to make something that should require little effort much too complex.

Section 2.3.1.1 documents the type of server that might be sufficient to serve CD-ROM images, or reference document files for network client use. This configuration is also discussed in Chapter 7, *Stand-alone Servers*, Section 7.3.1. The purpose for this configuration is to provide a shared volume that is read-only that anyone, even guests, can access.

The second example shows a minimal configuration for a print server that anyone can print to as long as they have the correct printer drivers installed on their computer. This is a

mirror of the system described in Chapter 7, *Stand-alone Servers*, Section 7.3.2.

The next example is of a secure office file and print server that will be accessible only to users who have an account on the system. This server is meant to closely resemble a Workgroup file and print server, but has to be more secure than an anonymous access machine. This type of system will typically suit the needs of a small office. The server does not provide network logon facilities, offers no Domain Control, instead it is just a network attached storage (NAS) device and a printserver.

Finally, we start looking at more complex systems that will either integrate into existing Microsoft Windows networks, or replace them entirely. The examples provided covers domain member servers as well as Samba Domain Control (PDC/BDC) and finally describes in detail a large distributed network with branch offices in remote locations.

2.3 Worked Examples

The configuration examples are designed to cover everything necessary to get Samba running. They do not cover basic operating system platform configuration, which is clearly beyond the scope of this text.

It is also assumed that Samba has been correctly installed, either by way of installation of the packages that are provided by the operating system vendor, or through other means.

2.3.1 Stand-alone Server

A Stand-alone Server implies no more than the fact that it is not a Domain Controller and it does not participate in Domain Control. It can be a simple workgroup-like server, or it may be a complex server that is a member of a domain security context.

2.3.1.1 Anonymous Read-Only Document Server

The purpose of this type of server is to make available to any user any documents or files that are placed on the shared resource. The shared resource could be a CD-ROM drive, a CD-ROM image, or a file storage area.

As the examples are developed, every attempt is made to progress the system toward greater capability, just as one might expect would happen in a real business office as that office grows in size and its needs change.

The configuration file is:

- The file system share point will be `/export`.
- All files will be owned by a user called Jack Baumbach. Jack's login name will be `jackb`. His password will be `m0r3pa1n` — of course, that's just the example we are using; do not use this in a production environment because all readers of this document will know it.

INSTALLATION PROCEDURE — READ-ONLY SERVER

1. Add user to system (with creation of the users' home directory):

Example 2.1. Anonymous Read-Only Server Configuration

```
# Global parameters
```

```
[global]
workgroup = MIDEARTH
netbios name = HOBBIT
security = share
```

```
[data]
comment = Data
path = /export
read only = No
guest only = Yes
```

```
root# useradd -c "Jack Baumbach" -m -g users -p m0r3pa1n jackb
```

2. Create directory, and set permissions and ownership:

```
root# mkdir /export
root# chmod u+rwx,g+rx,o+rx /export
root# chown jackb.users /export
```

3. Copy the files that should be shared to the `/export` directory.
4. Install the Samba configuration file (`/etc/samba/smb.conf`) as shown.
5. Test the configuration file:

```
root# testparm
```

Note any error messages that might be produced. Do not proceed until you obtain error-free output. An example of the output with the following file will list the file.

```
Load smb config files from /etc/samba/smb.conf
Processing section "[data]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
[Press enter]
```

```
# Global parameters
[global]
workgroup = MIDEARTH
```

```

netbios name = HOBBIT
security = share

[data]
comment = Data
path = /export
read only = No
guest only = Yes

```

6. Start Samba using the method applicable to your operating system platform.
7. Configure your Microsoft Windows client for workgroup *MIDEARTH*, set the machine name to *ROBBINS*, reboot, wait a few (2 - 5) minutes, then open Windows Explorer and visit the network neighborhood. The machine *HOBBIT* should be visible. When you click this machine icon, it should open up to reveal the *data* share. After clicking the share it, should open up to reveal the files previously placed in the */export* directory.

The information above (following # Global parameters) provides the complete contents of the */etc/samba/smb.conf* file.

2.3.1.2 Anonymous Read-Write Document Server

We should view this configuration as a progression from the previous example. The difference is that shared access is now forced to the user identity of *jackb* and to the primary group *jackb* belongs to. One other refinement we can make is to add the user *jackb* to the *smbpasswd* file. To do this execute:

```

root# smbpasswd -a jackb
New SMB password: m0r3pa1n
Retype new SMB password: m0r3pa1n
Added user jackb.

```

Addition of this user to the *smbpasswd* file allows all files to be displayed in the Explorer Properties boxes as belonging to *jackb* instead of to *User Unknown*.

The complete, modified *smb.conf* file is as shown in Example 2.2.

2.3.1.3 Anonymous Print Server

An anonymous print server serves two purposes:

- It allows printing to all printers from a single location.
- It reduces network traffic congestion due to many users trying to access a limited number of printers.

In the simplest of anonymous print servers, it is common to require the installation of the correct printer drivers on the Windows workstation. In this case the print server will be

Example 2.2. Modified Anonymous Read-Write smb.conf

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = HOBBIT
security = SHARE

[data]
comment = Data
path = /export
force user = jackb
force group = users
read only = No
guest ok = Yes
```

designed to just pass print jobs through to the spooler, and the spooler should be configured to do raw passthrough to the printer. In other words, the print spooler should not filter or process the data stream being passed to the printer.

In this configuration it is undesirable to present the Add Printer Wizard and we do not want to have automatic driver download, so we will disable it in the following configuration. Example 2.3 is the resulting `smb.conf` file.

Example 2.3. Anonymous Print Server smb.conf

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = LUTHIEN
security = share
printcap name = cups
disable spoolss = Yes
show add printer wizard = No
printing = cups

[printers]
comment = All Printers
path = /var/spool/samba
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = No
```

The above configuration is not ideal. It uses no smart features, and it deliberately presents a less than elegant solution. But it is basic, and it does print.

NOTE



Windows users will need to install a local printer and then change the print to device after installation of the drivers. The print to device can then be set to the network printer on this machine.

Make sure that the directory `/var/spool/samba` is capable of being used as intended. The following steps must be taken to achieve this:

- The directory must be owned by the superuser (`root`) user and group:

```
root# chown root.root /var/spool/samba
```

- Directory permissions should be set for public read-write with the sticky-bit set as shown:

```
root# chmod a+rwtX /var/spool/samba
```

NOTE



On CUPS enabled systems there is a facility to pass raw data directly to the printer without intermediate processing via CUPS print filters. Where use of this mode of operation is desired it is necessary to configure a raw printing device. It is also necessary to enable the raw mime handler in the `/etc/mime.conv` and `/etc/mime.types` files. Refer to Section 18.3.4.

2.3.1.4 Secure Read-Write File and Print Server

We progress now from simple systems to a server that is slightly more complex.

Our new server will require a public data storage area in which only authenticated users (i.e., those with a local account) can store files, as well as a home directory. There will be one printer that should be available for everyone to use.

In this hypothetical environment (no espionage was conducted to obtain this data), the site is demanding a simple environment that is *secure enough* but not too difficult to use.

Site users will be: Jack Baumbach, Mary Orville and Amed Sehkah. Each will have a password (not shown in further examples). Mary will be the printer administrator and will own all files in the public share.

This configuration will be based on *User Level Security* that is the default, and for which the default is to store Microsoft Windows-compatible encrypted passwords in a file called `/etc/samba/smbpasswd`. The default `smb.conf` entry that makes this happen is: `passwd backend = smbpasswd, guest`. Since this is the default it is not necessary to enter it into the configuration file. Note that guest backend is added to the list of active passwd backends not matter was it specified directly in Samba configuration file or not.

INSTALLING THE SECURE OFFICE SERVER

1. Add all users to the Operating System:

```
root# useradd -c "Jack Baumbach" -m -g users -p m0r3pa1n jackb
root# useradd -c "Mary Orville" -m -g users -p secret maryo
root# useradd -c "Amed Sehkah" -m -g users -p secret ameds
```

2. Configure the Samba `smb.conf` file as shown in Example 2.4.
3. Initialize the Microsoft Windows password database with the new users:

```
root# smbpasswd -a root
New SMB password: bigsecret
Reenter smb password: bigsecret
Added user root.
```

```
root# smbpasswd -a jackb
New SMB password: m0r3pa1n
Retype new SMB password: m0r3pa1n
Added user jackb.
```

```
root# smbpasswd -a maryo
New SMB password: secret
Reenter smb password: secret
Added user maryo.
```

```
root# smbpasswd -a ameds
New SMB password: mysecret
Reenter smb password: mysecret
Added user ameds.
```

4. Install printer using the CUPS Web interface. Make certain that all printers that will be shared with Microsoft Windows clients are installed as raw printing devices.
5. Start Samba using the operating system administrative interface. Alternately, this can be done manually by running:

```
root# nmbd; smbd;
```

Example 2.4. Secure Office Server smb.conf

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = OLORIN
printcap name = cups
disable spoolss = Yes
show add printer wizard = No
printing = cups

[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No

[public]
comment = Data
path = /export
force user = maryo
force group = users
guest ok = Yes

[printers]
comment = All Printers
path = /var/spool/samba
printer admin = root, maryo
create mask = 0600
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = No
```

6. Configure the /export directory:

```
root# mkdir /export
root# chown maryo.users /export
root# chmod u=rwx,g=rwx,o=rwx /export
```

7. Check that Samba is running correctly:

```
root# smbclient -L localhost -U%
Domain=[MIDEARTH] OS=[UNIX] Server=[Samba-3.0.0]
```

Sharename	Type	Comment
public	Disk	Data
IPC\$	IPC	IPC Service (Samba-3.0.0)
ADMIN\$	IPC	IPC Service (Samba-3.0.0)
hplj4	Printer	hplj4

Server	Comment
OLORIN	Samba-3.0.0

Workgroup	Master
MIDEARTH	OLORIN

8. Connect to OLORIN as maryo:

```

root# smbclient //olorin/maryo -Umaryo%secret
OS=[UNIX] Server=[Samba-3.0.0]
smb: \> dir
.                D           0   Sat Jun 21 10:58:16 2003
..               D           0   Sat Jun 21 10:54:32 2003
Documents        D           0   Fri Apr 25 13:23:58 2003
DOCWORK          D           0   Sat Jun 14 15:40:34 2003
OpenOffice.org   D           0   Fri Apr 25 13:55:16 2003
.bashrc          H          1286  Fri Apr 25 13:23:58 2003
.netscape6       DH           0   Fri Apr 25 13:55:13 2003
.mozilla         DH           0   Wed Mar  5 11:50:50 2003
.kermdrc         H           164  Fri Apr 25 13:23:58 2003
.acrobat         DH           0   Fri Apr 25 15:41:02 2003

55817 blocks of size 524288. 34725 blocks available
smb: \> q

```

By now you should be getting the hang of configuration basics. Clearly, it is time to explore slightly more complex examples. For the remainder of this chapter we will abbreviate instructions since there are previous examples.

2.3.2 Domain Member Server

In this instance we will consider the simplest server configuration we can get away with to make an accounting department happy. Let's be warned, the users are accountants and they do have some nasty demands. There is a budget for only one server for this department.

The network is managed by an internal Information Services Group (ISG), to which we belong. Internal politics are typical of a medium-sized organization; Human Resources is

of the opinion that they run the ISG because they are always adding and disabling users. Also, departmental managers have to fight tooth and nail to gain basic network resources access for their staff. Accounting is different though, they get exactly what they want. So this should set the scene.

We will use the users from the last example. The accounting department has a general printer that all departmental users may. There is also a check printer that may be used only by the person who has authority to print checks. The Chief Financial Officer (CFO) wants that printer to be completely restricted and for it to be located in the private storage area in her office. It therefore must be a network printer.

Accounting department uses an accounting application called *SpytFull* that must be run from a central application server. The software is licensed to run only off one server, there are no workstation components, and it is run off a mapped share. The data store is in a UNIX-based SQL backend. The UNIX gurus look after that, so is not our problem.

The accounting department manager (maryo) wants a general filing system as well as a separate file storage area for form letters (nastygrams). The form letter area should be read-only to all accounting staff except the manager. The general filing system has to have a structured layout with a general area for all staff to store general documents, as well as a separate file area for each member of her team that is private to that person, but she wants full access to all areas. Users must have a private home share for personal work-related files and for materials not related to departmental operations.

2.3.2.1 Example Configuration

The server *valinor* will be a member server of the company domain. Accounting will have only a local server. User accounts will be on the Domain Controllers as will desktop profiles and all network policy files.

1. Do not add users to the UNIX/Linux server; all of this will run off the central domain.
2. Configure `smb.conf` according to Example 2.5 and Example 2.6.
3. Join the domain. Note: Do not start Samba until this step has been completed!

```
root# net rpc join -Uroot%'bigsecret'  
Joined domain MIDEARTH.
```

4. Make absolutely certain that you disable (shut down) the `nscd` daemon on any system on which `winbind` is configured to run.
5. Start Samba following the normal method for your operating system platform. If you wish to this manually execute as root:

```
root# nmbd; smbd; winbindd;
```

6. Configure the name service switch control file on your system to resolve user and group names via winbind. Edit the following lines in `/etc/nsswitch.conf`:

Example 2.5. Member server smb.conf (globals)

```
# Global parameters
```

```
[global]
workgroup = MIDEARTH
netbios name = VALINOR
security = DOMAIN
printcap name = cups
disable spoolss = Yes
show add printer wizard = No
idmap uid = 15000-20000
idmap gid = 15000-20000
winbind separator = +
winbind use default domain = Yes
use sendfile = Yes
printing = cups
```

```
passwd: files winbind
group: files winbind
hosts: files dns winbind
```

7. Set the password for **wbinfo** to use:

```
root# wbinfo --set-auth-user=root%'bigsecret'
```

8. Validate that domain user and group credentials can be correctly resolved by executing:

```
root# wbinfo -u
MIDEARTH+maryo
MIDEARTH+jackb
MIDEARTH+ameds
...
MIDEARTH+root

root# wbinfo -g
MIDEARTH+Domain Users
MIDEARTH+Domain Admins
MIDEARTH+Domain Guests
...
MIDEARTH+Accounts
```

9. Check that **winbind** is working. The following demonstrates correct username resolution via the **getent** system utility:

Example 2.6. Member server smb.conf (shares and services)

```

[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No

[spytfull]
comment = Accounting Application Only
path = /export/spytfull
valid users = @Accounts
admin users = maryo
read only = Yes

[public]
comment = Data
path = /export/public
read only = No

[printers]
comment = All Printers
path = /var/spool/samba
printer admin = root, maryo
create mask = 0600
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = No

```

```

root# getent passwd maryo
maryo:x:15000:15003:Mary Orville:/home/MIDEARTH/maryo:/bin/false

```

10. A final test that we have this under control might be reassuring:

```

root# touch /export/a_file
root# chown maryo /export/a_file
root# ls -al /export/a_file
...
-rw-r--r--  1 maryo  users      11234 Jun 21 15:32 a_file
...

root# rm /export/a_file

```

11. Configuration is now mostly complete, so this is an opportune time to configure the directory structure for this site:

```
root# mkdir -p /export/{sptyfull,public}
root# chmod ug=rwxS,o=x /export/{sptyfull,public}
root# chown maryo.Accounts /export/{sptyfull,public}
```

2.3.3 Domain Controller

For the remainder of this chapter the focus is on the configuration of Domain Control. The examples that follow are for two implementation strategies. Remember, our objective is to create a simple but working solution. The remainder of this book should help to highlight opportunity for greater functionality and the complexity that goes with it.

A Domain Controller configuration can be achieved with a simple configuration using the new `tdbsam` password backend. This type of configuration is good for small offices, but has limited scalability (cannot be replicated) and performance can be expected to fall as the size and complexity of the domain increases.

The use of `tdbsam` is best limited to sites that do not need more than a primary Domain Controller (PDC). As the size of a domain grows the need for additional Domain Controllers becomes apparent. Do not attempt to under-resource a Microsoft Windows network environment; Domain Controllers provide essential authentication services. The following are symptoms of an under-resourced Domain Control environment:

- Domain logons intermittently fail.
- File access on a Domain Member server intermittently fails, giving a permission denied error message.

A more scalable Domain Control authentication backend option might use Microsoft Active Directory, or an LDAP-based backend. Samba-3 provides for both options as a Domain Member server. As a PDC Samba-3 is not able to provide an exact alternative to the functionality that is available with Active Directory. Samba-3 can provide a scalable LDAP-based PDC/BDC solution.

The `tdbsam` authentication backend provides no facility to replicate the contents of the database, except by external means. (i.e., there is no self-contained protocol in Samba-3 for Security Account Manager database [SAM] replication.)

NOTE



If you need more than one Domain Controller, do not use a `tdbsam` authentication backend.

2.3.3.1 Example: Engineering Office

The engineering office network server we present here is designed to demonstrate use of the new tdbsam password backend. The tdbsam facility is new to Samba-3. It is designed to provide many user and machine account controls that are possible with Microsoft Windows NT4. It is safe to use this in smaller networks.

1. A working PDC configuration using the tdbsam password backend can be found in Example 2.7 together with Example 2.8:

Example 2.7. Engineering Office smb.conf (globals)

```
[global]
workgroup = MIDEARTH
netbios name = FRODO
passdb backend = tdbsam
printcap name = cups
add user script = /usr/sbin/useradd -m %u
delete user script = /usr/sbin/userdel -r %u
add group script = /usr/sbin/groupadd %g
delete group script = /usr/sbin/groupdel %g
add user to group script = /usr/sbin/usermod -G %g %u
add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null %u
# Note: The following specifies the default logon script.
# Per user logon scripts can be specified in the user account using pdbedit
logon script = scripts\logon.bat
# This sets the default profile path. Set per user paths with pdbedit
logon path = \\%L\Profiles\%U
logon drive = H:
logon home = \\%L\%U
domain logons = Yes
os level = 35
preferred master = Yes
domain master = Yes
idmap uid = 15000-20000
idmap gid = 15000-20000
printing = cups
```

2. Create UNIX group accounts as needed using a suitable operating system tool:

```
root# groupadd ntadmins
root# groupadd designers
root# groupadd engineers
root# groupadd qateam
```

3. Create user accounts on the system using the appropriate tool provided with the operating system. Make sure all user home directories are created also. Add users to

Example 2.8. Eningeering Office smb.conf (shares and services)

```
[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No
# Printing auto-share (makes printers available thru CUPS)

[printers]
comment = All Printers
path = /var/spool/samba
printer admin = root, maryo
create mask = 0600
guest ok = Yes
printable = Yes
browseable = No

[print$]
comment = Printer Drivers Share
path = /var/lib/samba/drivers
write list = maryo, root
printer admin = maryo, root
# Needed to support domain logons

[netlogon]
comment = Network Logon Service
path = /var/lib/samba/netlogon
admin users = root, maryo
guest ok = Yes
browseable = No
# For profiles to work, create a user directory under the path
# shown. i.e., mkdir -p /var/lib/samba/profiles/maryo

[Profiles]
comment = Roaming Profile Share
path = /var/lib/samba/profiles
read only = No
profile acls = Yes
# Other resource (share/printer) definitions would follow below.
...
```

groups as required for access control on files, directories, printers, and as required for use in the Samba environment.

4. Assign each of the UNIX groups to NT groups: (It may be useful to copy this text to a shell script called `initGroups.sh`.)

[title] Shell script for initialising group mappings [/title]

```
#!/bin/bash
#### Keep this as a shell script for future re-use

# First assign well known groups
net groupmap modify ntgroup="Domain Admins" unixgroup=ntadmins rid=512
net groupmap modify ntgroup="Domain Users"  unixgroup=users      rid=513
net groupmap modify ntgroup="Domain Guests" unixgroup=nobody    rid=514

# Now for our added Domain Groups
net groupmap add ntgroup="Designers"  unixgroup=designers type=d rid=1112
net groupmap add ntgroup="Engineers"  unixgroup=engineers type=d rid=1113
net groupmap add ntgroup="QA Team"    unixgroup=qateam   type=d rid=1114
```

5. Create the `scripts` directory for use in the `[NETLOGON]` share:

```
root# mkdir -p /var/lib/samba/netlogon/scripts
```

Place the logon scripts that will be used (batch or cmd scripts) in this directory.

The above configuration provides a functional Primary Domain Control (PDC) system to which must be added file shares and printers as required.

2.3.3.2 A Big Organization

In this section we finally get to review in brief a Samba-3 configuration that uses a Light Weight Directory Access (LDAP)-based authentication backend. The main reasons for this choice are to provide the ability to host primary and Backup Domain Control (BDC), as well as to enable a higher degree of scalability to meet the needs of a very distributed environment.

The Primary Domain Controller This is an example of a minimal configuration to run a Samba-3 PDC using an LDAP authentication backend. It is assumed that the operating system has been correctly configured.

1. Obtain from the Samba sources `~/examples/LDAP/samba.schema` and copy it to the `/etc/openldap/schema/` directory.
2. Set up the LDAP server. This example is suitable for OpenLDAP 2.1.x. The `/etc/openldap/slapd.conf` file: [title] Example slapd.conf file [/title]

```
# Note commented out lines have been removed
include      /etc/openldap/schema/core.schema
include      /etc/openldap/schema/cosine.schema
include      /etc/openldap/schema/inetorgperson.schema
include      /etc/openldap/schema/nis.schema
```

```

include          /etc/openldap/schema/samba.schema

pidfile          /var/run/slapd/slapd.pid
argsfile         /var/run/slapd/slapd.args

database         bdb
suffix           "dc=kenya,dc=org"
rootdn           "cn=Manager,dc=kenya,dc=org"
rootpw           {SSHA}06qDkonA8hk6W6SSnRzWj0/pBcU3m0/P
# The password for the above is 'nastyon3'

directory       /var/lib/ldap

index  objectClass      eq
index  cn                pres,sub,eq
index  sn                pres,sub,eq
index  uid               pres,sub,eq
index  displayName       pres,sub,eq
index  uidNumber         eq
index  gidNumber         eq
index  memberUid         eq
index  sambaSID          eq
index  sambaPrimaryGroupSID eq
index  sambaDomainName   eq
index  default           sub

```

3. Create the following file `samba-ldap-init.ldif`:

```

# Organization for SambaXP Demo
dn: dc=kenya,dc=org
objectclass: dcObject
objectclass: organization
dc: kenya
o: SambaXP Demo
description: The SambaXP Demo LDAP Tree

# Organizational Role for Directory Management
dn: cn=Manager,dc=kenya,dc=org
objectclass: organizationalRole
cn: Manager
description: Directory Manager

# Setting up the container for users
dn: ou=People, dc=kenya, dc=org
objectclass: top
objectclass: organizationalUnit
ou: People

```

```
# Set up an admin handle for People OU
dn: cn=admin, ou=People, dc=kenya, dc=org
cn: admin
objectclass: top
objectclass: organizationalRole
objectclass: simpleSecurityObject
userPassword: {SSHA}0jBHgQ1vp4EDX2rEMMfIudvRMJoGwjVb
# The password for above is 'mordonL8'
```

4. Load the initial data above into the LDAP database:

```
root# slapadd -v -l initdb.ldif
```

5. Start the LDAP server using the appropriate tool or method for the operating system platform on which it is installed.
6. The `smb.conf` file that drives this backend can be found in example Example 2.9.
7. Add the LDAP password to the `secrets.tdc` file so Samba can update the LDAP database:

```
root# smbpasswd -w mordonL8
```

8. Add users and groups as required. Users and groups added using Samba tools will automatically be added to both the LDAP backend as well as to the operating system as required.

Backup Domain Controller Example 2.10 shows the example configuration for the BDC.

1. Decide if the BDC should have its own LDAP server or not. If the BDC is to be the LDAP server change the following `smb.conf` as indicated. The default configuration in Example 2.10 uses a central LDAP server.
2. Configure the NETLOGON and PROFILES directory as for the PDC in Example 2.10.

Example 2.9. LDAP backend smb.conf for PDC

Global parameters

```
[global]
workgroup = MIDEARTH
netbios name = FRODO
passdb backend = ldapsam:ldap://localhost
username map = /etc/samba/smbusers
printcap name = cups
add user script = /usr/sbin/useradd -m %u
delete user script = /usr/sbin/userdel -r %u
add group script = /usr/sbin/groupadd %g
delete group script = /usr/sbin/groupdel %g
add user to group script = /usr/sbin/usermod -G %g %u
add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null \
-g machines %u
logon script = scripts\logon.bat
logon path = \\%L\Profiles\%U
logon drive = H:
logon home = \\%L\%U
domain logons = Yes
os level = 35
preferred master = Yes
domain master = Yes
ldap suffix = dc=quenya,dc=org
ldap machine suffix = ou=People
ldap user suffix = ou=People
ldap group suffix = ou=People
ldap idmap suffix = ou=People
ldap admin dn = cn=Manager
ldap ssl = no
ldap passwd sync = Yes
idmap uid = 15000-20000
idmap gid = 15000-20000
winbind separator = +
printing = cups
...
```

Example 2.10. Remote LDAP BDC smb.conf

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = GANDALF
passdb backend = ldapsam:ldap://frodo.kenya.org
username map = /etc/samba/smbusers
printcap name = cups
add user script = /usr/sbin/useradd -m %u
delete user script = /usr/sbin/userdel -r %u
add group script = /usr/sbin/groupadd %g
delete group script = /usr/sbin/groupdel %g
add user to group script = /usr/sbin/usermod -G %g %u
add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null \
-g machines %u
logon script = scripts\logon.bat
logon path = \\%L\Profiles\%U
logon drive = H:
logon home = \\%L\%U
domain logons = Yes
os level = 33
preferred master = Yes
domain master = No
ldap suffix = dc=kenya,dc=org
ldap machine suffix = ou=People
ldap user suffix = ou=People
ldap group suffix = ou=People
ldap idmap suffix = ou=People
ldap admin dn = cn=Manager
ldap ssl = no
ldap passwd sync = Yes
idmap uid = 15000-20000
idmap gid = 15000-20000
winbind separator = +
printing = cups
...
```

Part II

Server Configuration Basics

SERVER TYPES AND SECURITY MODES

This chapter provides information regarding the types of server that Samba may be configured to be. A Microsoft network administrator who wishes to migrate to or use Samba will want to know the meaning, within a Samba context, of terms familiar to MS Windows administrator. This means that it is essential also to define how critical security modes function before we get into the details of how to configure the server itself.

The chapter provides an overview of the security modes of which Samba is capable and how they relate to MS Windows servers and clients.

A question often asked is, “*Why would I want to use Samba?*” Most chapters contain a section that highlights features and benefits. We hope that the information provided will help to answer this question. Be warned though, we want to be fair and reasonable, so not all features are positive towards Samba. The benefit may be on the side of our competition.

3.1 Features and Benefits

Two men were walking down a dusty road, when one suddenly kicked up a small red stone. It hurt his toe and lodged in his sandal. He took the stone out and cursed it with a passion and fury befitting his anguish. The other looked at the stone and said, “*This is a garnet. I can turn that into a precious gem and some day it will make a princess very happy!*”

The moral of this tale: Two men, two very different perspectives regarding the same stone. Like it or not, Samba is like that stone. Treat it the right way and it can bring great pleasure, but if you are forced to use it and have no time for its secrets, then it can be a source of discomfort.

Samba started out as a project that sought to provide interoperability for MS Windows 3.x clients with a UNIX server. It has grown up a lot since its humble beginnings and now provides features and functionality fit for large scale deployment. It also has some warts. In sections like this one we tell of both.

So, what are the benefits of features mentioned in this chapter?

- Samba-3 can replace an MS Windows NT4 Domain Controller.

- Samba-3 offers excellent interoperability with MS Windows NT4-style domains as well as natively with Microsoft Active Directory domains.
- Samba-3 permits full NT4-style Interdomain Trusts.
- Samba has security modes that permit more flexible authentication than is possible with MS Windows NT4 Domain Controllers.
- Samba-3 permits use of multiple account database backends.
- The account (password) database backends can be distributed and replicated using multiple methods. This gives Samba-3 greater flexibility than MS Windows NT4 and in many cases a significantly higher utility than Active Directory domains with MS Windows 200x.

3.2 Server Types

Administrators of Microsoft networks often refer to three different type of servers:

- Domain Controller
 - Primary Domain Controller
 - Backup Domain Controller
 - ADS Domain Controller
- Domain Member Server
 - Active Directory Domain Server
 - NT4 Style Domain Domain Server
- Stand-alone Server

The chapters covering Domain Control, Backup Domain Control and Domain Membership provide pertinent information regarding Samba configuration for each of these server roles. The reader is strongly encouraged to become intimately familiar with the information presented.

3.3 Samba Security Modes

In this section the function and purpose of Samba's security modes are described. An accurate understanding of how Samba implements each security mode as well as how to configure MS Windows clients for each mode will significantly reduce user complaints and administrator heartache.

In the SMB/CIFS networking world, there are only two types of security: *User Level* and *Share Level*. We refer to these collectively as *security levels*. In implementing these two security levels, Samba provides flexibilities that are not available with Microsoft Windows NT4/200x servers. In actual fact, Samba implements *Share Level* security only one way, but has four ways of implementing *User Level* security. Collectively, we call the Samba

implementations *Security Modes*. They are known as: *SHARE*, *USER*, *DOMAIN*, *ADS*, and *SERVER* modes. They are documented in this chapter.

An SMB server tells the client at startup what security level it is running. There are two options: Share Level and User Level. Which of these two the client receives affects the way the client then tries to authenticate itself. It does not directly affect (to any great extent) the way the Samba server does security. This may sound strange, but it fits in with the client/server approach of SMB. In SMB everything is initiated and controlled by the client, and the server can only tell the client what is available and whether an action is allowed.

3.3.1 User Level Security

We will describe User Level Security first, as its simpler. In User Level Security, the client will send a session setup request directly following protocol negotiation. This request provides a username and password. The server can either accept or reject that username/password combination. At this stage the server has no idea what share the client will eventually try to connect to, so it can't base the *accept/reject* on anything other than:

1. the username/password.
2. the name of the client machine.

If the server accepts the username/password then the client expects to be able to mount shares (using a *tree connection*) without specifying a password. It expects that all access rights will be as the username/password specified in the *session setup*.

It is also possible for a client to send multiple *session setup* requests. When the server responds, it gives the client a *uid* to use as an authentication tag for that username/password. The client can maintain multiple authentication contexts in this way (WinDD is an example of an application that does this).

3.3.1.1 Example Configuration

The `smb.conf` parameter that sets user level security is:

```
security = user
```

This is the default setting since Samba-2.2.x.

3.3.2 Share Level Security

In Share Level security, the client authenticates itself separately for each share. It sends a password along with each tree connection (share mount). It does not explicitly send a username with this operation. The client expects a password to be associated with each share, independent of the user. This means that Samba has to work out what username the client probably wants to use. It is never explicitly sent the username. Some commercial SMB servers such as NT actually associate passwords directly with shares in Share Level security, but Samba always uses the UNIX authentication scheme where it is a username/password pair that is authenticated, not a share/password pair.

To understand the MS Windows networking parallels, one should think in terms of MS Windows 9x/Me where one can create a shared folder that provides read-only or full access, with or without a password.

Many clients send a session setup even if the server is in Share Level security. They normally send a valid username but no password. Samba records this username in a list of possible usernames. When the client then does a tree connection it also adds to this list the name of the share they try to connect to (useful for home directories) and any users listed in the *user* parameter in the `smb.conf` file. The password is then checked in turn against these possible usernames. If a match is found then the client is authenticated as that user.

3.3.2.1 Example Configuration

The `smb.conf` parameter that sets Share Level security is:

```
security = share
```

There are reports that recent MS Windows clients do not like to work with share mode security servers. You are strongly discouraged from using Share Level security.

3.3.3 Domain Security Mode (User Level Security)

When Samba is operating in *security* = domain mode, the Samba server has a domain security trust account (a machine account) and causes all authentication requests to be passed through to the Domain Controllers. In other words, this configuration makes the Samba server a Domain Member server.

3.3.3.1 Example Configuration

Samba as a Domain Member Server

This method involves addition of the following parameters in the `smb.conf` file:

```
security = domain  
workgroup = MIDEARTH
```

In order for this method to work, the Samba server needs to join the MS Windows NT security domain. This is done as follows:

1. On the MS Windows NT Domain Controller, using the Server Manager, add a machine account for the Samba server.
2. On the UNIX/Linux system execute:

```
root# net rpc join -U administrator%password
```

NOTE

Samba-2.2.4 and later can auto-join a Windows NT4-style Domain just by executing:

```
root# smbpasswd -j DOMAIN_NAME -r PDC_NAME \  
-U Administrator%password
```



Samba-3 can do the same by executing:

```
root# net rpc join -U Administrator%password
```

It is not necessary with Samba-3 to specify the *DOMAIN_NAME* or the *PDC_NAME* as it figures this out from the *smb.conf* file settings.

Use of this mode of authentication does require there to be a standard UNIX account for each user in order to assign a UID once the account has been authenticated by the remote Windows DC. This account can be blocked to prevent logons by clients other than MS Windows through means such as setting an invalid shell in the */etc/passwd* entry.

An alternative to assigning UIDs to Windows users on a Samba member server is presented in Chapter 20, *Winbind: Use of Domain Accounts*.

For more information regarding Domain Membership, see Chapter 6, *Domain Membership*.

3.3.4 ADS Security Mode (User Level Security)

Both Samba-2.2, and Samba-3 can join an Active Directory domain. This is possible if the domain is run in native mode. Active Directory in native mode perfectly allows NT4-style Domain Members. This is contrary to popular belief. Active Directory in native mode prohibits only the use of Backup Domain Controllers running MS Windows NT4.

If you are using Active Directory, starting with Samba-3 you can join as a native AD member. Why would you want to do that? Your security policy might prohibit the use of NT-compatible authentication protocols. All your machines are running Windows 2000 and above and all use Kerberos. In this case Samba as an NT4-style domain would still require NT-compatible authentication data. Samba in AD-member mode can accept Kerberos tickets.

3.3.4.1 Example Configuration

```
realm = your.kerberos.REALM  
security = ADS
```

The following parameter may be required:

```
password server = your.kerberos.server
```

Please refer to Chapter 6, *Domain Membership* and Section 6.4 for more information regarding this configuration option.

3.3.5 Server Security (User Level Security)

Server Security Mode is left over from the time when Samba was not capable of acting as a Domain Member server. It is highly recommended not to use this feature. Server security mode has many drawbacks that include:

- Potential Account Lockout on MS Windows NT4/200x password servers.
- Lack of assurance that the password server is the one specified.
- Does not work with Winbind, which is particularly needed when storing profiles remotely.
- This mode may open connections to the password server, and keep them open for extended periods.
- Security on the Samba server breaks badly when the remote password server suddenly shuts down.
- With this mode there is NO security account in the domain that the password server belongs to for the Samba server.

In Server Security Mode the Samba server reports to the client that it is in User Level security. The client then does a session setup as described earlier. The Samba server takes the username/password that the client sends and attempts to login to the *password server* by sending exactly the same username/password that it got from the client. If that server is in User Level Security and accepts the password, then Samba accepts the client's connection. This allows the Samba server to use another SMB server as the *password server*.

You should also note that at the start of all this where the server tells the client what security level it is in, it also tells the client if it supports encryption. If it does, it supplies the client with a random cryptkey. The client will then send all passwords in encrypted form. Samba supports this type of encryption by default.

The parameter *security = server* means that Samba reports to clients that it is running in *user mode* but actually passes off all authentication requests to another *user mode* server. This requires an additional parameter *password server* that points to the real authentication server. The real authentication server can be another Samba server, or it can be a Windows NT server, the latter being natively capable of encrypted password support.

NOTE

When Samba is running in *Server Security Mode* it is essential that the parameter *password server* is set to the precise NetBIOS machine name of the target authentication server. Samba cannot determine this from NetBIOS name lookups because the choice of the target authentication server is arbitrary and cannot be determined from a domain name. In essence, a Samba server that is in *Server Security Mode* is operating in what used to be known as workgroup mode.

3.3.5.1 Example Configuration

Using MS Windows NT as an Authentication Server

This method involves the additions of the following parameters in the `smb.conf` file:

```
encrypt passwords = Yes
security = server
password server = "NetBIOS_name_of_a_DC"
```

There are two ways of identifying whether or not a username and password pair is valid. One uses the reply information provided as part of the authentication messaging process, the other uses just an error code.

The downside of this mode of configuration is the fact that for security reasons Samba will send the password server a bogus username and a bogus password and if the remote server fails to reject the username and password pair then an alternative mode of identification of validation is used. Where a site uses password lock out after a certain number of failed authentication attempts this will result in user lockouts.

Use of this mode of authentication requires a standard UNIX account for the user. This account can be blocked to prevent logons by non-SMB/CIFS clients.

3.4 Password Checking

MS Windows clients may use encrypted passwords as part of a challenge/response authentication model (a.k.a. NTLMv1 and NTLMv2) or alone, or cleartext strings for simple password-based authentication. It should be realized that with the SMB protocol, the password is passed over the network either in plain-text or encrypted, but not both in the same authentication request.

When encrypted passwords are used, a password that has been entered by the user is encrypted in two ways:

- An MD4 hash of the unicode of the password string. This is known as the NT hash.
- The password is converted to upper case, and then padded or truncated to 14 bytes. This string is then appended with 5 bytes of NULL characters and split to form two

56-bit DES keys to encrypt a “*magic*” 8-byte value. The resulting 16 bytes form the LanMan hash.

MS Windows 95 pre-service pack 1, MS Windows NT versions 3.x and version 4.0 pre-service pack 3 will use either mode of password authentication. All versions of MS Windows that follow these versions no longer support plain text passwords by default.

MS Windows clients have a habit of dropping network mappings that have been idle for 10 minutes or longer. When the user attempts to use the mapped drive connection that has been dropped, the client re-establishes the connection using a cached copy of the password.

When Microsoft changed the default password mode, support was dropped for caching of the plain-text password. This means that when the registry parameter is changed to re-enable use of plain-text passwords it appears to work, but when a dropped service connection mapping attempts to revalidate, this will fail if the remote authentication server does not support encrypted passwords. It is definitely not a good idea to re-enable plain-text password support in such clients.

The following parameters can be used to work around the issue of Windows 9x/Me clients upper-casing usernames and passwords before transmitting them to the SMB server when using cleartext authentication:

```
password level = integer  
username level = integer
```

By default Samba will convert to lower case the username before attempting to lookup the user in the database of local system accounts. Because UNIX usernames conventionally only contain lower-case characters, the *username level* parameter is rarely needed.

However, passwords on UNIX systems often make use of mixed-case characters. This means that in order for a user on a Windows 9x/Me client to connect to a Samba server using cleartext authentication, the *password level* must be set to the maximum number of upper case letters that *could* appear in a password. Note that if the server OS uses the traditional DES version of `crypt()`, a *password level* of 8 will result in case insensitive passwords as seen from Windows users. This will also result in longer login times as Samba has to compute the permutations of the password string and try them one by one until a match is located (or all combinations fail).

The best option to adopt is to enable support for encrypted passwords wherever Samba is used. Most attempts to apply the registry change to re-enable plain-text passwords will eventually lead to user complaints and unhappiness.

3.5 Common Errors

We all make mistakes. It is okay to make mistakes, as long as they are made in the right places and at the right time. A mistake that causes lost productivity is seldom tolerated, however a mistake made in a developmental test lab is expected.

Here we look at common mistakes and misapprehensions that have been the subject of discussions on the Samba mailing lists. Many of these are avoidable by doing your homework before attempting a Samba implementation. Some are the result of a misunderstanding of

the English language. The English language, which has many phrases that are potentially vague and may be highly confusing to those for whom English is not their native tongue.

3.5.1 What Makes Samba a Server?

To some the nature of the Samba *security* mode is obvious, but entirely wrong all the same. It is assumed that *security* = server means that Samba will act as a server. Not so! This setting means that Samba will *try* to use another SMB server as its source for user authentication alone.

3.5.2 What Makes Samba a Domain Controller?

The `smb.conf` parameter *security* = domain does not really make Samba behave as a Domain Controller. This setting means we want Samba to be a Domain Member.

3.5.3 What Makes Samba a Domain Member?

Guess! So many others do. But whatever you do, do not think that *security* = user makes Samba act as a Domain Member. Read the manufacturer's manual before the warranty expires. See Chapter 6, *Domain Membership* for more information.

3.5.4 Constantly Losing Connections to Password Server

“Why does server_validate() simply give up rather than re-establish its connection to the password server? Though I am not fluent in the SMB protocol, perhaps the cluster server process passes along to its client workstation the session key it receives from the password server, which means the password hashes submitted by the client would not work on a subsequent connection whose session key would be different. So server_validate() must give up.”

Indeed. That's why *security* = server is at best a nasty hack. Please use *security* = domain; *security* = server mode is also known as pass-through authentication.

DOMAIN CONTROL

There are many who approach MS Windows networking with incredible misconceptions. That's okay, because it gives the rest of us plenty of opportunity to be of assistance. Those who really want help would be well advised to become familiar with information that is already available.

The reader is advised not to tackle this section without having first understood and mastered some basics. MS Windows networking is not particularly forgiving of misconfiguration. Users of MS Windows networking are likely to complain of persistent niggles that may be caused by a broken network configuration. To a great many people, however, MS Windows networking starts with a Domain Controller that in some magical way is expected to solve all network operational ills.

The diagram in Figure 4.1 shows a typical MS Windows Domain Security network environment. Workstations A, B and C are representative of many physical MS Windows network clients.

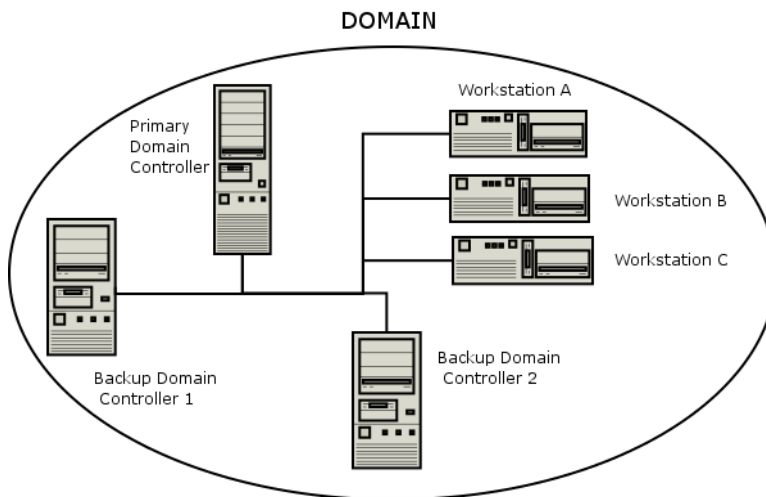


Figure 4.1. An Example Domain.

From the Samba mailing list one can readily identify many common networking issues. If you are not clear on the following subjects, then it will do much good to read the sections of this HOWTO that deal with it. These are the most common causes of MS Windows networking problems:

- Basic TCP/IP configuration.
- NetBIOS name resolution.
- Authentication configuration.
- User and group configuration.
- Basic file and directory permission control in UNIX/Linux.
- Understanding how MS Windows clients interoperate in a network environment.

Do not be put off; on the surface of it MS Windows networking seems so simple that anyone can do it. In fact, it is not a good idea to set up an MS Windows network with inadequate training and preparation. But let's get our first indelible principle out of the way: *It is perfectly okay to make mistakes!* In the right place and at the right time, mistakes are the essence of learning. It is very much not okay to make mistakes that cause loss of productivity and impose an avoidable financial burden on an organization.

Where is the right place to make mistakes? Only out of harm's way. If you are going to make mistakes, then please do it on a test network, away from users and in such a way as to not inflict pain on others. Do your learning on a test network.

4.1 Features and Benefits

What is the key benefit of Microsoft Domain Security?

In a word, *Single Sign On*, or SSO for short. To many, this is the Holy Grail of MS Windows NT and beyond networking. SSO allows users in a well-designed network to log onto any workstation that is a member of the domain that their user account is in (or in a domain that has an appropriate trust relationship with the domain they are visiting) and they will be able to log onto the network and access resources (shares, files and printers) as if they are sitting at their home (personal) workstation. This is a feature of the Domain Security protocols.

The benefits of Domain Security are available to those sites that deploy a Samba PDC. A Domain provides a unique network security identifier (SID). Domain user and group security identifiers are comprised of the network SID plus a relative identifier (RID) that is unique to the account. User and Group SIDs (the network SID plus the RID) can be used to create Access Control Lists (ACLs) attached to network resources to provide organizational access control. UNIX systems recognize only local security identifiers.

NOTE



Network clients of an MS Windows Domain Security Environment must be Domain Members to be able to gain access to the advanced features provided. Domain Membership involves more than just setting the workgroup name to the Domain name. It requires the creation of a Domain trust account for the workstation (called a machine account). Refer to Chapter 6, *Domain Membership* for more information.

The following functionalities are new to the Samba-3 release:

- Windows NT4 domain trusts.
- Adding users via the User Manager for Domains. This can be done on any MS Windows client using the `Nexus.exe` toolkit for Windows 9x/Me, or using the `SRV-TOOLS.EXE` package for MS Windows NT4/200x/XP platforms. These packages are available from Microsoft's Web site.
- Introduces replaceable and multiple user account (authentication) backends. In the case where the backend is placed in an LDAP database, Samba-3 confers the benefits of a backend that can be distributed, replicated and is highly scalable.
- Implements full Unicode support. This simplifies cross locale internationalization support. It also opens up the use of protocols that Samba-2.2.x had but could not use due to the need to fully support Unicode.

The following functionalities are not provided by Samba-3:

- SAM replication with Windows NT4 Domain Controllers (i.e., a Samba PDC and a Windows NT BDC or vice versa). This means Samba cannot operate as a BDC when the PDC is Microsoft-based or replicate account data to Windows BDCs.
- Acting as a Windows 2000 Domain Controller (i.e., Kerberos and Active Directory). In point of fact, Samba-3 does have some Active Directory Domain Control ability that is at this time purely experimental that is certain to change as it becomes a fully supported feature some time during the Samba-3 (or later) life cycle. However, Active Directory is more than just SMB — it's also LDAP, Kerberos, DHCP, and other protocols (with proprietary extensions, of course).
- The Windows 200x/XP MMC (Computer Management) Console can not be used to manage a Samba-3 server. For this you can use only the MS Windows NT4 Domain Server manager and the MS Windows NT4 Domain User Manager. Both are part of the `SVRTOOLS.EXE` package mentioned later.

Windows 9x/Me/XP Home clients are not true members of a domain for reasons outlined in this chapter. The protocol for support of Windows 9x/Me style network (domain) logons is completely different from NT4/Windows 200x type domain logons and has been officially supported for some time. These clients use the old LanMan Network Logon facilities that are supported in Samba since approximately the Samba-1.9.15 series.

Samba-3 implements group mapping between Windows NT groups and UNIX groups (this

is really quite complicated to explain in a short space). This is discussed more fully in Chapter 11, *Group Mapping MS Windows and UNIX*.

Samba-3, like an MS Windows NT4 PDC or a Windows 200x Active Directory, needs to store user and Machine Trust Account information in a suitable backend datastore. Refer to Section 6.2. With Samba-3 there can be multiple backends for this. A complete discussion of account database backends can be found in Chapter 10, *Account Information Databases*.

4.2 Basics of Domain Control

Over the years, public perceptions of what Domain Control really is has taken on an almost mystical nature. Before we branch into a brief overview of Domain Control, there are three basic types of Domain Controllers.

4.2.1 Domain Controller Types

- Primary Domain Controller
- Backup Domain Controller
- ADS Domain Controller

The *Primary Domain Controller* or PDC plays an important role in MS Windows NT4. In Windows 200x Domain Control architecture, this role is held by Domain Controllers. Folklore dictates that because of its role in the MS Windows network, the Domain Controller should be the most powerful and most capable machine in the network. As strange as it may seem to say this here, good overall network performance dictates that the entire infrastructure needs to be balanced. It is advisable to invest more in Stand-alone (Domain Member) servers than in the Domain Controllers.

In the case of MS Windows NT4-style domains, it is the PDC that initiates a new Domain Control database. This forms a part of the Windows registry called the Security Account Manager (SAM). It plays a key part in NT4-type domain user authentication and in synchronization of the domain authentication database with Backup Domain Controllers.

With MS Windows 200x Server-based Active Directory domains, one Domain Controller initiates a potential hierarchy of Domain Controllers, each with their own area of delegated control. The master domain controller has the ability to override any downstream controller, but a downline controller has control only over its downline. With Samba-3, this functionality can be implemented using an LDAP-based user and machine account backend.

New to Samba-3 is the ability to use a backend database that holds the same type of data as the NT4-style SAM database (one of the registry files)¹.

The *Backup Domain Controller* or BDC plays a key role in servicing network authentication requests. The BDC is biased to answer logon requests in preference to the PDC. On a network segment that has a BDC and a PDC, the BDC will most likely service network logon requests. The PDC will answer network logon requests when the BDC is too busy (high load). A BDC can be promoted to a PDC. If the PDC is online at the time that a BDC

¹See also Chapter 10, *Account Information Databases*.

is promoted to PDC, the previous PDC is automatically demoted to a BDC. With Samba-3, this is not an automatic operation; the PDC and BDC must be manually configured and changes also need to be made.

With MS Windows NT4, a decision is made at installation to determine what type of machine the server will be. It is possible to promote a BDC to a PDC and vice versa. The only way to convert a Domain Controller to a Domain Member server or a Stand-alone Server is to reinstall it. The install time choices offered are:

- *Primary Domain Controller* — the one that seeds the domain SAM.
- *Backup Domain Controller* — one that obtains a copy of the domain SAM.
- *Domain Member Server* — one that has no copy of the domain SAM, rather it obtains authentication from a Domain Controller for all access controls.
- *Stand-alone Server* — one that plays no part in SAM synchronization, has its own authentication database and plays no role in Domain Security.

With MS Windows 2000, the configuration of Domain Control is done after the server has been installed. Samba-3 is capable of acting fully as a native member of a Windows 200x server Active Directory domain.

New to Samba-3 is the ability to function fully as an MS Windows NT4-style Domain Controller, excluding the SAM replication components. However, please be aware that Samba-3 also supports the MS Windows 200x Domain Control protocols.

At this time any appearance that Samba-3 is capable of acting as an *Domain Controller* in native ADS mode is limited and experimental in nature. This functionality should not be used until the Samba Team offers formal support for it. At such a time, the documentation will be revised to duly reflect all configuration and management requirements. Samba can act as a NT4-style DC in a Windows 2000/XP environment. However, there are certain compromises:

- No machine policy files.
- No Group Policy Objects.
- No synchronously executed AD logon scripts.
- Can't use Active Directory management tools to manage users and machines.
- Registry changes tattoo the main registry, while with AD they do not leave permanent changes in effect.
- Without AD you cannot perform the function of exporting specific applications to specific users or groups.

4.2.2 Preparing for Domain Control

There are two ways that MS Windows machines may interact with each other, with other servers and with Domain Controllers: either as *Stand-alone* systems, more commonly called *Workgroup* members, or as full participants in a security system, more commonly called *Domain* members.

It should be noted that *Workgroup* membership involves no special configuration other than the machine being configured so the network configuration has a commonly used name for its workgroup entry. It is not uncommon for the name WORKGROUP to be used for this. With this mode of configuration, there are no Machine Trust Accounts and any concept of membership as such is limited to the fact that all machines appear in the network neighborhood to be logically grouped together. Again, just to be clear: *workgroup mode does not involve security machine accounts*.

Domain Member machines have a machine account in the Domain accounts database. A special procedure must be followed on each machine to effect Domain Membership. This procedure, which can be done only by the local machine Administrator account, will create the Domain machine account (if it does not exist), and then initializes that account. When the client first logs onto the Domain it triggers a machine password change.

NOTE



When Samba is configured as a Domain Controller, secure network operation demands that all MS Windows NT4/200x/XP Professional clients should be configured as Domain Members. If a machine is not made a member of the Domain, then it will operate like a workgroup (Stand-alone) machine. Please refer to Chapter 6, *Domain Membership* for information regarding Domain Membership.

The following are necessary for configuring Samba-3 as an MS Windows NT4-style PDC for MS Windows NT4/200x/XP clients:

- Configuration of basic TCP/IP and MS Windows networking.
- Correct designation of the Server Role (*security* = user).
- Consistent configuration of Name Resolution².
- Domain logons for Windows NT4/200x/XP Professional clients.
- Configuration of Roaming Profiles or explicit configuration to force local profile usage.
- Configuration of network/system policies.
- Adding and managing domain user accounts.
- Configuring MS Windows client machines to become Domain Members.

The following provisions are required to serve MS Windows 9x/Me clients:

- Configuration of basic TCP/IP and MS Windows networking.
- Correct designation of the server role (*security* = user).
- Network Logon Configuration (since Windows 9x/Me/XP Home are not technically domain members, they do not really participate in the security aspects of Domain logons as such).

²See Chapter 9, *Network Browsing*, and Chapter 25, *Integrating MS Windows Networks with Samba*.

- Roaming Profile Configuration.
- Configuration of System Policy handling.
- Installation of the network driver “*Client for MS Windows Networks*” and configuration to log onto the domain.
- Placing Windows 9x/Me clients in User Level Security — if it is desired to allow all client share access to be controlled according to domain user/group identities.
- Adding and managing domain user accounts.

NOTE



Roaming Profiles and System/Network policies are advanced network administration topics that are covered in the Chapter 23, *Desktop Profile Management* and Chapter 22, *System and Account Policies* chapters of this document. However, these are not necessarily specific to a Samba PDC as much as they are related to Windows NT networking concepts.

A Domain Controller is an SMB/CIFS server that:

- Registers and advertises itself as a Domain Controller (through NetBIOS broadcasts as well as by way of name registrations either by Mailslot Broadcasts over UDP broadcast, to a WINS server over UDP unicast, or via DNS and Active Directory).
- Provides the NETLOGON service. (This is actually a collection of services that runs over multiple protocols. These include the LanMan Logon service, the Netlogon service, the Local Security Account service, and variations of them.)
- Provides a share called NETLOGON.

It is rather easy to configure Samba to provide these. Each Samba Domain Controller must provide the NETLOGON service that Samba calls the *domain logons* functionality (after the name of the parameter in the `smb.conf` file). Additionally, one server in a Samba-3 Domain must advertise itself as the Domain Master Browser³. This causes the Primary Domain Controller to claim a domain-specific NetBIOS name that identifies it as a Domain Master Browser for its given domain or workgroup. Local master browsers in the same domain or workgroup on broadcast-isolated subnets then ask for a complete copy of the browse list for the whole wide area network. Browser clients will then contact their Local Master Browser, and will receive the domain-wide browse list, instead of just the list for their broadcast-isolated subnet.

4.3 Domain Control — Example Configuration

The first step in creating a working Samba PDC is to understand the parameters necessary in `smb.conf`. An example `smb.conf` for acting as a PDC can be found in Example 4.1.

³See Chapter 9, *Network Browsing*.

Example 4.1. smb.conf for being a PDC

```

[global]
netbios name = BELERIAND
workgroup = MIDEARTH
passdb backend = tdbsam
os level = 33
preferred master = yes
domain master = yes
local master = yes
security = user
domain logons = yes
logon path = \\%N\profiles\%u
logon drive = H:
logon home = \\homeserver\%u\winprofile
logon script = logon.cmd

[netlogon]
path = /var/lib/samba/netlogon
read only = yes
write list = ntadmin

[profiles]
path = /var/lib/samba/profiles
read only = no
create mask = 0600
directory mask = 0700

```

The basic options shown in Example 4.1 are explained as follows:

passdb backend — This contains all the user and group account information. Acceptable values for a PDC are: *smbpasswd*, *tdbsam*, and *ldapsam*. The “*guest*” entry provides default accounts and is included by default, there is no need to add it explicitly.

Where use of backup Domain Controllers (BDCs) is intended, the only logical choice is to use LDAP so the *passdb* backend can be distributed. The *tdbsam* and *smbpasswd* files cannot effectively be distributed and therefore should not be used.

Domain Control Parameters — The parameters *os level*, *preferred master*, *domain master*, *security*, *encrypt passwords*, and *domain logons* play a central role in assuring domain control and network logon support.

The *os level* must be set at or above a value of 32. A Domain Controller must be the Domain Master Browser, must be set in *user* mode security, must support Microsoft-compatible encrypted passwords, and must provide the network logon service (domain logons). Encrypted passwords must be enabled. For more details on how to do this,

refer to Chapter 10, *Account Information Databases*.

Environment Parameters — The parameters *logon path*, *logon home*, *logon drive*, and *logon script* are environment support settings that help to facilitate client logon operations and that help to provide automated control facilities to ease network management overheads. Please refer to the man page information for these parameters.

NETLOGON Share — The NETLOGON share plays a central role in domain logon and Domain Membership support. This share is provided on all Microsoft Domain Controllers. It is used to provide logon scripts, to store Group Policy files (NT-Config.POL), as well as to locate other common tools that may be needed for logon processing. This is an essential share on a Domain Controller.

PROFILE Share — This share is used to store user desktop profiles. Each user must have a directory at the root of this share. This directory must be write-enabled for the user and must be globally read-enabled. Samba-3 has a VFS module called “*fake_permissions*” that may be installed on this share. This will allow a Samba administrator to make the directory read-only to everyone. Of course this is useful only after the profile has been properly created.

NOTE

The above parameters make for a full set of parameters that may define the server's mode of operation. The following `smb.conf` parameters are the essentials alone:



```
netbios name = BELERIAND
workgroup = MIDEARTH
domain logons = Yes
domain master = Yes
security = User
```

The additional parameters shown in the longer listing above just makes for a more complete explanation.

4.4 Samba ADS Domain Control

Samba-3 is not, and cannot act as, an Active Directory Server. It cannot truly function as an Active Directory Primary Domain Controller. The protocols for some of the functionality of Active Directory Domain Controllers has been partially implemented on an experimental only basis. Please do not expect Samba-3 to support these protocols. Do not depend on any such functionality either now or in the future. The Samba Team may remove these experimental features or may change their behavior. This is mentioned for the benefit of

those who have discovered secret capabilities in Samba-3 and who have asked when this functionality will be completed. The answer is maybe or maybe never!

To be sure, Samba-3 is designed to provide most of the functionality that Microsoft Windows NT4-style Domain Controllers have. Samba-3 does not have all the capabilities of Windows NT4, but it does have a number of features that Windows NT4 domain controllers do not have. In short, Samba-3 is not NT4 and it is not Windows Server 200x, it is not an Active Directory server. We hope this is plain and simple enough for all to understand.

4.5 Domain and Network Logon Configuration

The subject of Network or Domain Logons is discussed here because it forms an integral part of the essential functionality that is provided by a Domain Controller.

4.5.1 Domain Network Logon Service

All Domain Controllers must run the netlogon service (*domain logons* in Samba). One Domain Controller must be configured with *domain master* = Yes (the Primary Domain Controller); on all Backup Domain Controllers *domain master* = No must be set.

4.5.1.1 Example Configuration

Example 4.2. smb.conf for being a PDC

```
[global]
domain logons = Yes
domain master = (Yes on PDC, No on BDCs)

[netlogon]
comment = Network Logon Service
path = /var/lib/samba/netlogon
guest ok = Yes
browseable = No
```

4.5.1.2 The Special Case of MS Windows XP Home Edition

To be completely clear: If you want MS Windows XP Home Edition to integrate with your MS Windows NT4 or Active Directory Domain Security, understand it cannot be done. The only option is to purchase the upgrade from MS Windows XP Home Edition to MS Windows XP Professional.

NOTE



MS Windows XP Home Edition does not have the ability to join any type of Domain Security facility. Unlike MS Windows 9x/Me, MS Windows XP Home Edition also completely lacks the ability to log onto a network.

Now that this has been said, please do not ask the mailing list or email any of the Samba Team members with your questions asking how to make this work. It can't be done. If it can be done, then to do so would violate your software license agreement with Microsoft, and we recommend that you do not do that.

4.5.1.3 The Special Case of Windows 9x/Me

A domain and a workgroup are exactly the same in terms of network browsing. The difference is that a distributable authentication database is associated with a domain, for secure login access to a network. Also, different access rights can be granted to users if they successfully authenticate against a domain logon server. Samba-3 does this now in the same way as MS Windows NT/200x.

The SMB client logging on to a domain has an expectation that every other server in the domain should accept the same authentication information. Network browsing functionality of domains and workgroups is identical and is explained in this documentation under the browsing discussions. It should be noted that browsing is totally orthogonal to logon support.

Issues related to the single-logon network model are discussed in this section. Samba supports domain logons, network logon scripts and user profiles for MS Windows for workgroups and MS Windows 9X/ME clients, which are the focus of this section.

When an SMB client in a domain wishes to logon, it broadcasts requests for a logon server. The first one to reply gets the job, and validates its password using whatever mechanism the Samba administrator has installed. It is possible (but ill advised) to create a domain where the user database is not shared between servers, i.e., they are effectively workgroup servers advertising themselves as participating in a domain. This demonstrates how authentication is quite different from but closely involved with domains.

Using these features you can make your clients verify their logon via the Samba server; make clients run a batch file when they logon to the network and download their preferences, desktop and start menu.

MS Windows XP Home edition is not able to join a domain and does not permit the use of domain logons.

Before launching into the configuration instructions, it is worthwhile to look at how a Windows 9x/Me client performs a logon:

1. The client broadcasts (to the IP broadcast address of the subnet it is in) a NetLogon request. This is sent to the NetBIOS name DOMAIN<#1c> at the NetBIOS layer.

The client chooses the first response it receives, which contains the NetBIOS name of the logon server to use in the format of `\\SERVER`.

2. The client connects to that server, logs on (does an `SMBsessetupX`) and then connects to the `IPC$` share (using an `SMBtconX`).
3. The client does a `NetWkstaUserLogon` request, which retrieves the name of the user's logon script.
4. The client then connects to the `NetLogon` share and searches for said script. If it is found and can be read, it is retrieved and executed by the client. After this, the client disconnects from the `NetLogon` share.
5. The client sends a `NetUserGetInfo` request to the server to retrieve the user's home share, which is used to search for profiles. Since the response to the `NetUserGetInfo` request does not contain much more than the user's home share, profiles for Windows 9x clients must reside in the user home directory.
6. The client connects to the user's home share and searches for the user's profile. As it turns out, you can specify the user's home share as a sharename and path. For example, `\\server\\fred\\.winprofile`. If the profiles are found, they are implemented.
7. The client then disconnects from the user's home share and reconnects to the `NetLogon` share and looks for `CONFIG.POL`, the policies file. If this is found, it is read and implemented.

The main difference between a PDC and a Windows 9x/Me logon server configuration is:

- Password encryption is not required for a Windows 9x/Me logon server. But note that beginning with MS Windows 98 the default setting is that plain-text password support is disabled. It can be re-enabled with the registry changes that are documented in Chapter 22, *System and Account Policies*.
- Windows 9x/Me clients do not require and do not use Machine Trust Accounts.

A Samba PDC will act as a Windows 9x/Me logon server; after all, it does provide the network logon services that MS Windows 9x/Me expect to find.

NOTE



Use of plain-text passwords is strongly discouraged. Where used they are easily detected using a sniffer tool to examine network traffic.

4.5.2 Security Mode and Master Browsers

There are a few comments to make in order to tie up some loose ends. There has been much debate over the issue of whether it is okay to configure Samba as a Domain Controller in security modes other than user. The only security mode that will not work due to technical

reasons is share-mode security. Domain and server mode security are really just a variation on SMB User Level Security.

Actually, this issue is also closely tied to the debate on whether Samba must be the Domain Master Browser for its workgroup when operating as a DC. While it may technically be possible to configure a server as such (after all, browsing and domain logons are two distinctly different functions), it is not a good idea to do so. You should remember that the DC must register the DOMAIN<#1b> NetBIOS name. This is the name used by Windows clients to locate the DC. Windows clients do not distinguish between the DC and the DMB. A DMB is a Domain Master Browser — see Section 9.4.1. For this reason, it is wise to configure the Samba DC as the DMB.

Now back to the issue of configuring a Samba DC to use a mode other than *security = user*. If a Samba host is configured to use another SMB server or DC in order to validate user connection requests, it is a fact that some other machine on the network (the *password server*) knows more about the user than the Samba host. About 99% of the time, this other host is a Domain Controller. Now to operate in domain mode security, the *workgroup* parameter must be set to the name of the Windows NT domain (which already has a Domain Controller). If the domain does not already have a Domain Controller, you do not yet have a Domain.

Configuring a Samba box as a DC for a domain that already by definition has a PDC is asking for trouble. Therefore, you should always configure the Samba DC to be the DMB for its domain and set *security = user*. This is the only officially supported mode of operation.

4.6 Common Errors

4.6.1 “\$” Cannot Be Included in Machine Name

A machine account, typically stored in `/etc/passwd`, takes the form of the machine name with a “\$” appended. FreeBSD (and other BSD systems) will not create a user with a “\$” in the name.

The problem is only in the program used to make the entry. Once made, it works perfectly. Create a user without the “\$”. Then use **vipw** to edit the entry, adding the “\$”. Or create the whole entry with **vipw** if you like; make sure you use a unique user login ID.

NOTE



The machine account must have the exact name that the workstation has.

NOTE



The UNIX tool **vipw** is a common tool for directly editing the `/etc/passwd` file.

4.6.2 Joining Domain Fails Because of Existing Machine Account

“I get told, ‘You already have a connection to the Domain...’ or ‘Cannot join domain, the credentials supplied conflict with an existing set...’ when creating a Machine Trust Account.”

This happens if you try to create a Machine Trust Account from the machine itself and already have a connection (e.g., mapped drive) to a share (or IPC\$) on the Samba PDC. The following command will remove all network drive connections:

```
C:\> net use * /d
```

Further, if the machine is already a “*member of a workgroup*” that is the same name as the domain you are joining (bad idea) you will get this message. Change the workgroup name to something else, it does not matter what, reboot, and try again.

4.6.3 The System Cannot Log You On (C000019B)

“I joined the domain successfully but after upgrading to a newer version of the Samba code I get the message, ‘The system cannot log you on (C000019B), Please try again or consult your system administrator when attempting to logon.”

This occurs when the domain SID stored in the `secrets.tdb` database is changed. The most common cause of a change in domain SID is when the domain name and/or the server name (NetBIOS name) is changed. The only way to correct the problem is to restore the original domain SID or remove the domain client from the domain and rejoin. The domain SID may be reset using either the `net` or `rpeclient` utilities.

To reset or change the domain SID you can use the `net` command as follows:

```
root# net getlocalsid 'OLDNAME'  
root# net setlocalsid 'SID'
```

Workstation Machine Trust Accounts work only with the Domain (or network) SID. If this SID changes Domain Members (workstations) will not be able to log onto the domain. The original Domain SID can be recovered from the `secrets.tdb` file. The alternative is to visit each workstation to re-join it to the domain.

4.6.4 The Machine Trust Account Is Not Accessible

“When I try to join the domain I get the message, ‘The machine account for this computer either does not exist or is not accessible’. What’s wrong?”

This problem is caused by the PDC not having a suitable Machine Trust Account. If you are using the *add machine script* method to create accounts then this would indicate that it has not worked. Ensure the domain admin user system is working.

Alternately, if you are creating account entries manually then they have not been created correctly. Make sure that you have the entry correct for the Machine Trust Account in `smbpasswd` file on the Samba PDC. If you added the account using an editor rather than using the `smbpasswd` utility, make sure that the account name is the machine NetBIOS name with a “\$” appended to it (i.e., `computer_name$`). There must be an entry in both `/etc/passwd` and the `smbpasswd` file.

Some people have also reported that inconsistent subnet masks between the Samba server and the NT client can cause this problem. Make sure that these are consistent for both client and server.

4.6.5 Account Disabled

“When I attempt to login to a Samba Domain from a NT4/W200x workstation, I get a message about my account being disabled.”

Enable the user accounts with `smbpasswd -e username`. This is normally done as an account is created.

4.6.6 Domain Controller Unavailable

“Until a few minutes after Samba has started, clients get the error ‘Domain Controller Unavailable’”

A Domain Controller has to announce its role on the network. This usually takes a while. Be patient for up to fifteen minutes, then try again.

4.6.7 Cannot Log onto Domain Member Workstation After Joining Domain

After successfully joining the domain, user logons fail with one of two messages: one to the effect that the Domain Controller cannot be found; the other claims that the account does not exist in the domain or that the password is incorrect. This may be due to incompatible settings between the Windows client and the Samba-3 server for *schannel* (secure channel) settings or *smb signing* settings. Check your Samba settings for *client schannel*, *server schannel*, *client signing*, *server signing* by executing:

```
testparm -v | more and looking for the value of these parameters.
```

Also use the Microsoft Management Console — Local Security Settings. This tool is available from the Control Panel. The Policy settings are found in the Local Policies/Security Options area and are prefixed by *Secure Channel: ...*, and *Digitally sign*

It is important that these be set consistently with the Samba-3 server settings.

BACKUP DOMAIN CONTROL

Before you continue reading this section, please make sure that you are comfortable with configuring a Samba Domain Controller as described in Chapter 4, *Domain Control*.

5.1 Features and Benefits

This is one of the most difficult chapters to summarize. It does not matter what we say here for someone will still draw conclusions and/or approach the Samba Team with expectations that are either not yet capable of being delivered, or that can be achieved far more effectively using a totally different approach. In the event that you should have a persistent concern that is not addressed in this book, please email John H. Terpstra¹ clearly setting out your requirements and/or question and we will do our best to provide a solution.

Samba-3 is capable of acting as a Backup Domain Controller (BDC) to another Samba Primary Domain Controller (PDC). A Samba-3 PDC can operate with an LDAP Account backend. The LDAP backend can be either a common master LDAP server, or a slave server. The use of a slave LDAP server has the benefit that when the master is down, clients may still be able to log onto the network. This effectively gives Samba a high degree of scalability and is an effective solution for large organizations. If you use an LDAP slave server for a PDC, you will need to ensure the master's continued availability - if the slave finds it's master down at the wrong time, you will have stability and operational problems.

While it is possible to run a Samba-3 BDC with non-LDAP backend, that backend must allow some form of 'two way' propagation, of changes from the BDC to the master. Only LDAP is capable of this at this stage.

The use of a non-LDAP backend SAM database is particularly problematic because Domain Member servers and workstations periodically change the Machine Trust Account password. The new password is then stored only locally. This means that in the absence of a centrally stored accounts database (such as that provided with an LDAP-based solution) if Samba-3 is running as a BDC, the BDC instance of the Domain Member trust account password will not reach the PDC (master) copy of the SAM. If the PDC SAM is then replicated to BDCs, this results in overwriting the SAM that contains the updated (changed) trust account password with resulting breakage of the domain trust.

¹<mailto:jht@samba.org>

Considering the number of comments and questions raised concerning how to configure a BDC, let's consider each possible option and look at the pros and cons for each possible solution. Table 5.1 lists possible design configurations for a PDC/BDC infrastructure.

Table 5.1. Domain Backend Account Distribution Options

PDC Backend	BDC Backend	Notes/Discussion
Master LDAP Server	Slave LDAP Server	The optimal solution that provides high integrity. The SAM will be replicated to a common master LDAP server.
Single Central LDAP Server	Single Central LDAP Server	A workable solution without fail-over ability. This is a useable solution, but not optimal.
tdbsam	tdbsam + net rpc vampire	Does not work with Samba-3.0.0; may be implemented in a later release. The downside of this solution is that an external process will control account database integrity. This solution may appeal to sites that wish to avoid the complexity of LDAP. The net rpc vampire is used to synchronize domain accounts from the PDC to the BDC.
tdbsam	tdbsam + rsync	Do not use this configuration. Does not work because the TDB files are live and data may not have been flushed to disk. Use rsync to synchronize the TDB database files from the PDC to the BDC.
smbpasswd file	smbpasswd file	Do not use this configuration. Not an elegant solution due to the delays in synchronization. Use rsync to synchronize the TDB database files from the PDC to the BDC. Can be made to work using a cron job to synchronize data from the PDC to the BDC.

5.2 Essential Background Information

A Domain Controller is a machine that is able to answer logon requests from network workstations. Microsoft LanManager and IBM LanServer were two early products that provided this capability. The technology has become known as the LanMan Netlogon service.

When MS Windows NT3.10 was first released, it supported a new style of Domain Control and with it a new form of the network logon service that has extended functionality. This service became known as the NT NetLogon Service. The nature of this service has changed with the evolution of MS Windows NT and today provides a complex array of services that are implemented over an intricate spectrum of technologies.

5.2.1 MS Windows NT4-style Domain Control

Whenever a user logs into a Windows NT4/200x/XP Professional Workstation, the workstation connects to a Domain Controller (authentication server) to validate that the username and password the user entered are valid. If the information entered does not match account information that has been stored in the Domain Control database (the SAM, or Security Account Manager database), a set of error codes is returned to the workstation that has made the authentication request.

When the username/password pair has been validated, the Domain Controller (authentication server) will respond with full enumeration of the account information that has been stored regarding that user in the User and Machine Accounts database for that Domain. This information contains a complete network access profile for the user but excludes any information that is particular to the user's desktop profile, or for that matter it excludes all desktop profiles for groups that the user may belong to. It does include password time limits, password uniqueness controls, network access time limits, account validity information, machine names from which the user may access the network, and much more. All this information was stored in the SAM in all versions of MS Windows NT (3.10, 3.50, 3.51, 4.0).

The account information (user and machine) on Domain Controllers is stored in two files, one containing the Security information and the other the SAM. These are stored in files by the same name in the `C:\Windows NT\System32\config` directory. These are the files that are involved in replication of the SAM database where Backup Domain Controllers are present on the network.

There are two situations in which it is desirable to install Backup Domain Controllers:

- On the local network that the Primary Domain Controller is on, if there are many workstations and/or where the PDC is generally very busy. In this case the BDCs will pick up network logon requests and help to add robustness to network services.
- At each remote site, to reduce wide area network traffic and to add stability to remote network operations. The design of the network, the strategic placement of Backup Domain Controllers, together with an implementation that localizes as much of network to client interchange as possible will help to minimize wide area network bandwidth needs (and thus costs).

The inter-operation of a PDC and its BDCs in a true Windows NT4 environment is worth mentioning here. The PDC contains the master copy of the SAM. In the event that an administrator makes a change to the user account database while physically present on the local network that has the PDC, the change will likely be made directly to the PDC instance of the master copy of the SAM. In the event that this update may be performed in a branch office, the change will likely be stored in a delta file on the local BDC. The BDC will then send a trigger to the PDC to commence the process of SAM synchronization. The PDC will then request the delta from the BDC and apply it to the master SAM. The PDC will then contact all the BDCs in the Domain and trigger them to obtain the update and then apply that to their own copy of the SAM.

Samba-3 can not participate in true SAM replication and is therefore not able to employ precisely the same protocols used by MS Windows NT4. A Samba-3 BDC will not create

SAM update delta files. It will not inter-operate with a PDC (NT4 or Samba) to synchronize the SAM from delta files that are held by BDCs.

Samba-3 cannot function as a BDC to an MS Windows NT4 PDC, and Samba-3 can not function correctly as a PDC to an MS Windows NT4 BDC. Both Samba-3 and MS Windows NT4 can function as a BDC to its own type of PDC.

The BDC is said to hold a *read-only* of the SAM from which it is able to process network logon requests and authenticate users. The BDC can continue to provide this service, particularly while, for example, the wide area network link to the PDC is down. A BDC plays a very important role in both the maintenance of Domain Security as well as in network integrity.

In the event that the NT4 PDC should need to be taken out of service, or if it dies, one of the NT4 BDCs can be promoted to a PDC. If this happens while the original NT4 PDC is on line, it is automatically demoted to an NT4 BDC. This is an important aspect of Domain Controller management. The tool that is used to effect a promotion or a demotion is the Server Manager for Domains. It should be noted that Samba-3 BDCs can not be promoted in this manner because reconfiguration of Samba requires changes to the `smb.conf` file.

5.2.1.1 Example PDC Configuration

Beginning with Version 2.2, Samba officially supports domain logons for all current Windows clients, including Windows NT4, 2003 and XP Professional. For Samba to be enabled as a PDC, some parameters in the *[global]*-section of the `smb.conf` have to be set. Refer to Example 5.1 for an example of the minimum required settings.

Example 5.1. Minimal `smb.conf` for a PDC in Use With a BDC LDAP Server on PDC.

```
workgroup = MIDEARTH
passdb backend = ldapsam://localhost:389
domain master = yes
domain logons = yes
```

Several other things like a *[homes]* and a *[netlogon]* share also need to be set along with settings for the profile path, the user's home drive, and so on. This is not covered in this chapter; for more information please refer to Chapter 4, *Domain Control*.

5.2.2 LDAP Configuration Notes

When configuring a master and a slave LDAP server, it is advisable to use the master LDAP server for the PDC and slave LDAP servers for the BDCs. It is not essential to use slave LDAP servers, however, many administrators will want to do so in order to provide redundant services. Of course, one or more BDCs may use any slave LDAP server. Then again, it is entirely possible to use a single LDAP server for the entire network.

When configuring a master LDAP server that will have slave LDAP servers, do not forget to configure this in the `/etc/openldap/slapd.conf` file. It must be noted that the DN of a server certificate must use the CN attribute to name the server, and the CN must carry the servers' fully qualified domain name. Additional alias names and wildcards may be present

in the subjectAltName certificate extension. More details on server certificate names are in RFC2830.

It does not really fit within the scope of this document, but a working LDAP installation is basic to LDAP enabled Samba operation. When using an OpenLdap server with Transport Layer Security (TLS), the machine name in `/etc/ssl/certs/slaped.pem` must be the same as in `/etc/openldap/slapd.conf`. The Red Hat Linux startup script creates the `slaped.pem` file with hostname `"localhost.localdomain."` It is impossible to access this LDAP server from a slave LDAP server (i.e., a Samba BDC) unless the certificate is recreated with a correct hostname.

Do not install a Samba PDC on a OpenLDAP slave server. Joining client machines to the domain will fail in this configuration because the change to the machine account in the LDAP tree must take place on the master LDAP server. This is not replicated rapidly enough to the slave server that the PDC queries. It therefore gives an error message on the client machine about not being able to set up account credentials. The machine account is created on the LDAP server but the password fields will be empty.

Possible PDC/BDC plus LDAP configurations include:

- PDC+BDC -> One Central LDAP Server.
- PDC -> LDAP master server, BDC -> LDAP slave server.
- PDC -> LDAP master, with secondary slave LDAP server.
BDC -> LDAP master, with secondary slave LDAP server.
- PDC -> LDAP master, with secondary slave LDAP server.
BDC -> LDAP slave server, with secondary master LDAP server.

In order to have a fall-back configuration (secondary) LDAP server one would specify the secondary LDAP server in the `smb.conf` file as shown in Example 5.2.

Example 5.2. Multiple LDAP Servers in `smb.conf`

```
...  
passdb backend = ldapsam:"ldap:ldap://master.quenya.org \  
ldapsam:ldap://slave.quenya.org  
...
```

5.2.3 Active Directory Domain Control

As of the release of MS Windows 2000 and Active Directory, this information is now stored in a directory that can be replicated and for which partial or full administrative control can be delegated. Samba-3 is not able to be a Domain Controller within an Active Directory tree, and it cannot be an Active Directory server. This means that Samba-3 also cannot act as a Backup Domain Controller to an Active Directory Domain Controller.

5.2.4 What Qualifies a Domain Controller on the Network?

Every machine that is a Domain Controller for the domain MIDEARTH has to register the NetBIOS group name MIDEARTH<#1c> with the WINS server and/or by broadcast on the local network. The PDC also registers the unique NetBIOS name MIDEARTH<#1b> with the WINS server. The name type <#1b> name is normally reserved for the Domain Master Browser, a role that has nothing to do with anything related to authentication, but the Microsoft Domain implementation requires the Domain Master Browser to be on the same machine as the PDC.

Where a WINS server is not used, broadcast name registrations alone must suffice. Refer to Section 9.3 for more information regarding TCP/IP network protocols and how SMB/CIFS names are handled.

5.2.5 How does a Workstation find its Domain Controller?

There are two different mechanisms to locate a domain controller, one method is used when NetBIOS over TCP/IP is enabled and the other when it has been disabled in the TCP/IP network configuration.

Where NetBIOS over TCP/IP is disabled, all name resolution involves the use of DNS, broadcast messaging over UDP, as well as Active Directory communication technologies. In this type of environment all machines require appropriate DNS entries. More information may be found in Section 9.3.3.

5.2.5.1 NetBIOS Over TCP/IP Enabled

An MS Windows NT4/200x/XP Professional workstation in the domain MIDEARTH that wants a local user to be authenticated has to find the Domain Controller for MIDEARTH. It does this by doing a NetBIOS name query for the group name MIDEARTH<#1c>. It assumes that each of the machines it gets back from the queries is a Domain Controller and can answer logon requests. To not open security holes, both the workstation and the selected Domain Controller authenticate each other. After that the workstation sends the user's credentials (name and password) to the local Domain Controller for validation.

5.2.5.2 NetBIOS Over TCP/IP Disabled

An MS Windows NT4/200x/XP Professional workstation in the realm `quenia.org` that has a need to affect user logon authentication will locate the Domain Controller by requesting DNS servers for the `_ldap._tcp.pdc.ms-dcs.quenia.org` record. More information regarding this subject may be found in Section 9.3.3.

5.3 Backup Domain Controller Configuration

The creation of a BDC requires some steps to prepare the Samba server before `smbd` is executed for the first time. These steps are outlined as follows:

- The domain SID has to be the same on the PDC and the BDC. In Samba versions pre-2.2.5, the domain SID was stored in the file `private/MACHINE.SID`. The domain SID is now stored in the file `private/secrets.tdb`. This file is unique to each server and can not be copied from a PDC to a BDC, the BDC will generate a new SID at start-up. It will over-write the PDC domain SID with the newly created BDC SID. There is a procedure that will allow the BDC to acquire the Domain SID. This is described here.

To retrieve the domain SID from the PDC or an existing BDC and store it in the `secrets.tdb`, execute:

```
root# net rpc getsid
```

- Specification of the `ldap admin dn` is obligatory. This also requires the LDAP administration password to be set in the `secrets.tdb` using the `smbpasswd -w mysecret`.
- Either `ldap suffix` or `ldap idmap suffix` must be specified in the `smb.conf` file.
- The UNIX user database has to be synchronized from the PDC to the BDC. This means that both the `/etc/passwd` and `/etc/group` have to be replicated from the PDC to the BDC. This can be done manually whenever changes are made. Alternately, the PDC is set up as an NIS master server and the BDC as an NIS slave server. To set up the BDC as a mere NIS client would not be enough, as the BDC would not be able to access its user database in case of a PDC failure. NIS is by no means the only method to synchronize passwords. An LDAP solution would also work.
- The Samba password database must be replicated from the PDC to the BDC. Although it is possible to synchronize the `smbpasswd` file with `rsync` and `ssh`, this method is broken and flawed, and is therefore not recommended. A better solution is to set up slave LDAP servers for each BDC and a master LDAP server for the PDC.
- The netlogon share has to be replicated from the PDC to the BDC. This can be done manually whenever login scripts are changed, or it can be done automatically using a `cron` job that will replicate the directory structure in this share using a tool like `rsync`.

5.3.1 Example Configuration

Finally, the BDC has to be found by the workstations. This can be done by setting Samba as shown in Example 5.3.

Example 5.3. Minimal setup for being a BDC

```
workgroup = MIDEARTH
passdb backend = ldapsam:ldap://slave-ldap.kenya.org
domain master = no
domain logons = yes
idmap backend = ldap:ldap://slave-ldap.kenya.org
```

In the *[global]*-section of the `smb.conf` of the BDC. This makes the BDC only register the name `SAMBA<#1c>` with the WINS server. This is no problem as the name `SAMBA<#1c>` is a NetBIOS group name that is meant to be registered by more than one machine. The parameter `domain master = no` forces the BDC not to register `SAMBA<#1b>` which as a unique NetBIOS name is reserved for the Primary Domain Controller.

The `idmap backend` will redirect the `winbindd` utility to use the LDAP database to resolve all UIDs and GIDs for UNIX accounts.

NOTE



Samba-3 has introduced a new ID mapping facility. One of the features of this facility is that it allows greater flexibility in how user and group IDs are handled in respect to NT Domain User and Group SIDs. One of the new facilities provides for explicitly ensuring that UNIX/Linux UID and GID values will be consistent on the PDC, all BDCs and all Domain Member servers. The parameter that controls this is called `idmap backend`. Please refer to the man page for `smb.conf` for more information regarding its behavior.

The use of the `idmap backend = ldap:ldap://master.kenya.org` option on a BDC only make sense where `ldapsam` is used on a PDC. The purpose for an LDAP based `idmap backend` is also to allow a domain-member (without its own `passdb backend`) to use `winbindd` to resolve Windows network users and groups to common UID/GIDs. In other words, this option is generally intended for use on BDCs and on Domain Member servers.

5.4 Common Errors

As this is a rather new area for Samba, there are not many examples that we may refer to. Updates will be published as they become available and may be found in later Samba releases or from the Samba web site.²

5.4.1 Machine Accounts Keep Expiring

This problem will occur when the `passdb (SAM)` files are copied from a central server but the local Backup Domain Controller is acting as a PDC. This results in the application of Local Machine Trust Account password updates to the local SAM. Such updates are not copied back to the central server. The newer machine account password is then over written when the SAM is re-copied from the PDC. The result is that the Domain Member machine on start up will find that its passwords do not match the one now in the database and since the startup security check will now fail, this machine will not allow logon attempts to proceed and the account expiry error will be reported.

²<http://samba.org>

The solution is to use a more robust passdb backend, such as the ldapsam backend, setting up a slave LDAP server for each BDC, and a master LDAP server for the PDC.

5.4.2 Can Samba Be a Backup Domain Controller to an NT4 PDC?

No. The native NT4 SAM replication protocols have not yet been fully implemented.

Can I get the benefits of a BDC with Samba? Yes, but only to a Samba PDC. The main reason for implementing a BDC is availability. If the PDC is a Samba machine, a second Samba machine can be set up to service logon requests whenever the PDC is down.

5.4.3 How Do I Replicate the smbpasswd File?

Replication of the smbpasswd file is sensitive. It has to be done whenever changes to the SAM are made. Every user's password change is done in the smbpasswd file and has to be replicated to the BDC. So replicating the smbpasswd file very often is necessary.

As the smbpasswd file contains plain text password equivalents, it must not be sent unencrypted over the wire. The best way to set up smbpasswd replication from the PDC to the BDC is to use the utility rsync. rsync can use ssh as a transport. **ssh** itself can be set up to accept *only* **rsync** transfer without requiring the user to type a password.

As said a few times before, use of this method is broken and flawed. Machine trust accounts will go out of sync, resulting in a broken domain. This method is *not* recommended. Try using LDAP instead.

5.4.4 Can I Do This All with LDAP?

The simple answer is yes. Samba's pdb_ldap code supports binding to a replica LDAP server, and will also follow referrals and rebind to the master if it ever needs to make a modification to the database. (Normally BDCs are read only, so this will not occur often).

DOMAIN MEMBERSHIP

Domain Membership is a subject of vital concern. Samba must be able to participate as a member server in a Microsoft Domain Security context, and Samba must be capable of providing Domain machine member trust accounts, otherwise it would not be able to offer a viable option for many users.

This chapter covers background information pertaining to Domain Membership, the Samba configuration for it, and MS Windows client procedures for joining a domain. Why is this necessary? Because both are areas in which there exists within the current MS Windows networking world and particularly in the UNIX/Linux networking and administration world, a considerable level of misinformation, incorrect understanding and a lack of knowledge. Hopefully this chapter will fill the voids.

6.1 Features and Benefits

MS Windows workstations and servers that want to participate in Domain Security need to be made Domain Members. Participating in Domain Security is often called *Single Sign On* or SSO for short. This chapter describes the process that must be followed to make a workstation (or another server — be it an MS Windows NT4 / 200x server) or a Samba server a member of an MS Windows Domain Security context.

Samba-3 can join an MS Windows NT4-style domain as a native member server, an MS Windows Active Directory Domain as a native member server, or a Samba Domain Control network. Domain Membership has many advantages:

- MS Windows workstation users get the benefit of SSO.
- Domain user access rights and file ownership/access controls can be set from the single Domain Security Account Manager (SAM) database (works with Domain Member servers as well as with MS Windows workstations that are Domain Members).
- Only MS Windows NT4/200x/XP Professional workstations that are Domain Members can use network logon facilities.
- Domain Member workstations can be better controlled through the use of Policy files (NTConfig.POL) and Desktop Profiles.

- Through the use of logon scripts, users can be given transparent access to network applications that run off application servers.
- Network administrators gain better application and user access management abilities because there is no need to maintain user accounts on any network client or server, other than the central Domain database (either NT4/Samba SAM style Domain, NT4 Domain that is backed up with an LDAP directory, or via an Active Directory infrastructure).

6.2 MS Windows Workstation/Server Machine Trust Accounts

A Machine Trust Account is an account that is used to authenticate a client machine (rather than a user) to the Domain Controller server. In Windows terminology, this is known as a “*Computer Account*.” The purpose of the machine account is to prevent a rogue user and Domain Controller from colluding to gain access to a domain member workstation.

The password of a Machine Trust Account acts as the shared secret for secure communication with the Domain Controller. This is a security feature to prevent an unauthorized machine with the same NetBIOS name from joining the domain and gaining access to domain user/group accounts. Windows NT/200x/XP Professional clients use machine trust accounts, but Windows 9x/Me/XP Home clients do not. Hence, a Windows 9x/Me/XP Home client is never a true member of a Domain because it does not possess a Machine Trust Account, and, thus, has no shared secret with the Domain Controller.

A Windows NT4 PDC stores each Machine Trust Account in the Windows Registry. The introduction of MS Windows 2000 saw the introduction of Active Directory, the new repository for Machine Trust Accounts. A Samba PDC, however, stores each Machine Trust Account in two parts, as follows:

- A Domain Security Account (stored in the *passwd backend* that has been configured in the *smb.conf* file. The precise nature of the account information that is stored depends on the type of backend database that has been chosen.

The older format of this data is the *smbpasswd* database that contains the UNIX login ID, the UNIX user identifier (UID), and the LanMan and NT encrypted passwords. There is also some other information in this file that we do not need to concern ourselves with here.

The two newer database types are called *ldapsam*, and *tdbsam*. Both store considerably more data than the older *smbpasswd* file did. The extra information enables new user account controls to be implemented.

- A corresponding UNIX account, typically stored in */etc/passwd*. Work is in progress to allow a simplified mode of operation that does not require UNIX user accounts, but this may not be a feature of the early releases of Samba-3.

There are three ways to create Machine Trust Accounts:

- Manual creation from the UNIX/Linux command line. Here, both the Samba and corresponding UNIX account are created by hand.

- Using the MS Windows NT4 Server Manager, either from an NT4 Domain Member server, or using the Nexus toolkit available from the Microsoft Web site. This tool can be run from any MS Windows machine as long as the user is logged on as the administrator account.
- “*On-the-fly*” creation. The Samba Machine Trust Account is automatically created by Samba at the time the client is joined to the domain. (For security, this is the recommended method.) The corresponding UNIX account may be created automatically or manually.

6.2.1 Manual Creation of Machine Trust Accounts

The first step in manually creating a Machine Trust Account is to manually create the corresponding UNIX account in `/etc/passwd`. This can be done using `vipw` or another “*add user*” command that is normally used to create new UNIX accounts. The following is an example for a Linux-based Samba server:

```
root# /usr/sbin/useradd -g machines -d /dev/null -c "machine nickname" \
    -s /bin/false machine_name$

root# passwd -l machine_name$
```

In the above example above there is an existing system group “*machines*” which is used as the primary group for all machine accounts. In the following examples the “*machines*” group has numeric GID equal 100.

On *BSD systems, this can be done using the `chpass` utility:

```
root# chpass -a \
'machine_name$:*:101:100::0:0:Windows machine_name:/dev/null:/sbin/nologin'
```

The `/etc/passwd` entry will list the machine name with a “\$” appended, will not have a password, will have a null shell and no home directory. For example, a machine named “*doppy*” would have an `/etc/passwd` entry like this:

```
doppy$:x:505:100:machine_nickname:/dev/null:/bin/false
```

Above, *machine_nickname* can be any descriptive name for the client, i.e., BasementComputer. *machine_name* absolutely must be the NetBIOS name of the client to be joined to the domain. The “\$” must be appended to the NetBIOS name of the client or Samba will not recognize this as a Machine Trust Account.

Now that the corresponding UNIX account has been created, the next step is to create the Samba account for the client containing the well-known initial Machine Trust Account password. This can be done using the `smbpasswd` command as shown here:

```
root# smbpasswd -a -m machine_name
```

where *machine_name* is the machine's NetBIOS name. The RID of the new machine account is generated from the UID of the corresponding UNIX account.

JOIN THE CLIENT TO THE DOMAIN IMMEDIATELY



Manually creating a Machine Trust Account using this method is the equivalent of creating a Machine Trust Account on a Windows NT PDC using the Server Manager. From the time at which the account is created to the time the client joins the domain and changes the password, your domain is vulnerable to an intruder joining your domain using a machine with the same NetBIOS name. A PDC inherently trusts members of the domain and will serve out a large degree of user information to such clients. You have been warned!

6.2.2 Managing Domain Machine Accounts using NT4 Server Manager

A working *add machine script* script is essential for machine trust accounts to be automatically created. This applies no matter whether one uses automatic account creation, or if one wishes to use the NT4 Domain Server Manager.

If the machine from which you are trying to manage the domain is an MS Windows NT4 workstation or MS Windows 200x/XP Professional, the tool of choice is the package called **SRVTOOLS.EXE**. When executed in the target directory it will unpack **SrvMgr.exe** and **UsrMgr.exe** (both are domain management tools for MS Windows NT4 workstation).

If your workstation is a Microsoft Windows 9x/Me family product you should download the **Nexus.exe** package from the Microsoft web site. When executed from the target directory this will unpack the same tools but for use on this platform.

Further information about these tools may be obtained from the following locations:

<http://support.microsoft.com/default.aspx?scid=kb;en-us;173673>

<http://support.microsoft.com/default.aspx?scid=kb;en-us;172540>

Launch the **srvmgr.exe** (Server Manager for Domains) and follow these steps:

SERVER MANAGER ACCOUNT MACHINE ACCOUNT MANAGEMENT

1. From the menu select **Computer**.
2. Click **Select Domain**.
3. Click the name of the domain you wish to administer in the **Select Domain** panel and then click **OK**.
4. Again from the menu select **Computer**.

5. Select **Add to Domain**.
6. In the dialog box, click the radio button to **Add NT Workstation of Server**, then enter the machine name in the field provided, and click the **Add** button.

6.2.3 On-the-Fly Creation of Machine Trust Accounts

The second (and recommended) way of creating Machine Trust Accounts is simply to allow the Samba server to create them as needed when the client is joined to the domain.

Since each Samba Machine Trust Account requires a corresponding UNIX account, a method for automatically creating the UNIX account is usually supplied; this requires configuration of the add machine script option in `smb.conf`. This method is not required, however, corresponding UNIX accounts may also be created manually.

Here is an example for a Red Hat Linux system.

```
add machine script = /usr/sbin/useradd -d /dev/null -g 100 \  
-s /bin/false -M %u
```

6.2.4 Making an MS Windows Workstation or Server a Domain Member

The procedure for making an MS Windows workstation or server a member of the domain varies with the version of Windows.

6.2.4.1 Windows 200x/XP Professional Client

When the user elects to make the client a Domain Member, Windows 200x prompts for an account and password that has privileges to create machine accounts in the domain. A Samba Administrator Account (i.e., a Samba account that has root privileges on the Samba server) must be entered here; the operation will fail if an ordinary user account is given.

For security reasons, the password for this Administrator Account should be set to a password that is other than that used for the root user in `/etc/passwd`.

The name of the account that is used to create Domain Member machine accounts can be anything the network administrator may choose. If it is other than `root` then this is easily mapped to `root` in the file named in the `smb.conf` parameter `username map = /etc/samba/smbusers`.

The session key of the Samba Administrator Account acts as an encryption key for setting the password of the machine trust account. The Machine Trust Account will be created on-the-fly, or updated if it already exists.

6.2.4.2 Windows NT4 Client

If the Machine Trust Account was created manually, on the Identification Changes menu enter the domain name, but do not check the box **Create a Computer Account in the Domain**. In this case, the existing Machine Trust Account is used to join the machine to the domain.

If the Machine Trust Account is to be created on-the-fly, on the Identification Changes menu enter the domain name and check the box **Create a Computer Account in the Domain**. In this case, joining the domain proceeds as above for Windows 2000 (i.e., you must supply a Samba Administrator Account when prompted).

6.2.4.3 Samba Client

Joining a Samba client to a domain is documented in Section 6.3.

6.3 Domain Member Server

This mode of server operation involves the Samba machine being made a member of a domain security context. This means by definition that all user authentication will be done from a centrally defined authentication regime. The authentication regime may come from an NT3/4-style (old domain technology) server, or it may be provided from an Active Directory server (ADS) running on MS Windows 2000 or later.

Of course it should be clear that the authentication backend itself could be from any distributed directory architecture server that is supported by Samba. This can be LDAP (from OpenLDAP), or Sun's iPlanet, or NetWare Directory Server, and so on.

NOTE



When Samba is configured to use an LDAP, or other identity management and/or directory service, it is Samba that continues to perform user and machine authentication. It should be noted that the LDAP server does not perform authentication handling in place of what Samba is designed to do.

Please refer to Chapter 4, *Domain Control*, for more information regarding how to create a domain machine account for a Domain Member server as well as for information on how to enable the Samba Domain Member machine to join the domain and be fully trusted by it.

6.3.1 Joining an NT4-type Domain with Samba-3

Table 6.1 lists names that have been used in the remainder of this chapter.

Table 6.1. Assumptions

NetBIOS name:	SERV1
Windows 200x/NT domain name:	MIDEARTH
Domain's PDC NetBIOS name:	DOMPDC
Domain's BDC NetBIOS names:	DOMBDC1 and DOMBDC2

First, you must edit your `smb.conf` file to tell Samba it should now use domain security.

Change (or add) your *security* line in the [global] section of your `smb.conf` to read:

```
security = domain
```

Next change the *workgroup* line in the [global] section to read:

```
workgroup = MIDEARTH
```

This is the name of the domain we are joining.

You must also have the parameter *encrypt passwords* set to *yes* in order for your users to authenticate to the NT PDC. This is the default setting if this parameter is not specified. There is no need to specify this parameter, but if it is specified in the `smb.conf` file, it must be set to *Yes*.

Finally, add (or modify) a *password server* line in the [global] section to read:

```
password server = DOMPDC DOMBDC1 DOMBDC2
```

These are the primary and backup Domain Controllers Samba will attempt to contact in order to authenticate users. Samba will try to contact each of these servers in order, so you may want to rearrange this list in order to spread out the authentication load among Domain Controllers.

Alternately, if you want `smbd` to automatically determine the list of Domain Controllers to use for authentication, you may set this line to be:

```
password server = *
```

This method allows Samba to use exactly the same mechanism that NT does. The method either uses broadcast-based name resolution, performs a WINS database lookup in order to find a Domain Controller against which to authenticate, or locates the Domain Controller using DNS name resolution.

To join the domain, run this command:

```
root# net join -S DOMPDC -UAdministrator%password
```

If the `-S DOMPDC` argument is not given, the domain name will be obtained from `smb.conf`.

The machine is joining the domain `DOM`, and the PDC for that domain (the only machine that has write access to the domain SAM database) is `DOMPDC`, therefore use the `-S` option. The `Administrator%password` is the login name and password for an account that has the necessary privilege to add machines to the domain. If this is successful, you will see the message in your terminal window the text shown below. Where the older NT4 style domain architecture is used:

```
Joined domain DOM.
```

Where Active Directory is used:

```
Joined SERV1 to realm MYREALM.
```

Refer to the **net** man page for further information.

This process joins the server to the domain without having to create the machine trust account on the PDC beforehand.

This command goes through the machine account password change protocol, then writes the new (random) machine account password for this Samba server into a file in the same directory in which a smbpasswd file would be normally stored:

```
/usr/local/samba/private/secrets.tdb  
or  
/etc/samba/secrets.tdb.
```

This file is created and owned by root and is not readable by any other user. It is the key to the Domain-level security for your system, and should be treated as carefully as a shadow password file.

Finally, restart your Samba daemons and get ready for clients to begin using domain security. The way you can restart your Samba daemons depends on your distribution, but in most cases the following will suffice:

```
root# /etc/init.d/samba restart
```

6.3.2 Why Is This Better Than `security = server`?

Currently, domain security in Samba does not free you from having to create local UNIX users to represent the users attaching to your server. This means that if Domain user `DOM\fred` attaches to your Domain Security Samba server, there needs to be a local UNIX user `fred` to represent that user in the UNIX file system. This is similar to the older Samba security mode `security = server`, where Samba would pass through the authentication request to a Windows NT server in the same way as a Windows 95 or Windows 98 server would.

Please refer to Chapter 20, *Winbind: Use of Domain Accounts*, for information on a system to automatically assign UNIX UIDs and GIDs to Windows NT Domain users and groups.

The advantage to Domain-level security is that the authentication in Domain-level security is passed down the authenticated RPC channel in exactly the same way that an NT server would do it. This means Samba servers now participate in domain trust relationships in exactly the same way NT servers do (i.e., you can add Samba servers into a resource domain and have the authentication passed on from a resource domain PDC to an account domain PDC).

In addition, with `security = server`, every Samba daemon on a server has to keep a connection open to the authenticating server for as long as that daemon lasts. This can

drain the connection resources on a Microsoft NT server and cause it to run out of available connections. With *security* = domain, however, the Samba daemons connect to the PDC/BDC only for as long as is necessary to authenticate the user and then drop the connection, thus conserving PDC connection resources.

And finally, acting in the same manner as an NT server authenticating to a PDC means that as part of the authentication reply, the Samba server gets the user identification information such as the user SID, the list of NT groups the user belongs to, and so on.

NOTE



Much of the text of this document was first published in the Web magazine LinuxWorld^a as the article <http://www.linuxworld.com/linuxworld/lw-1998-10/lw-10-samba.html>^b *Doing the NIS/NT Samba*.

^a<http://www.linuxworld.com>

^b<http://www.linuxworld.com/linuxworld/lw-1998-10/lw-10-samba.html>

6.4 Samba ADS Domain Membership

This is a rough guide to setting up Samba-3 with Kerberos authentication against a Windows 200x KDC. A familiarity with Kerberos is assumed.

6.4.1 Configure smb.conf

You must use at least the following three options in `smb.conf`:

```
realm = your.kerberos.REALM
security = ADS
encrypt passwords = yes
```

In case samba cannot correctly identify the appropriate ADS server using the realm name, use the *password server* option in `smb.conf`:

```
password server = your.kerberos.server
```

NOTE



You do *not* need a `smbpasswd` file, and older clients will be authenticated as if *security* = domain, although it will not do any harm and allows you to have local users not in the domain.

6.4.2 Configure /etc/krb5.conf

With both MIT and Heimdal Kerberos, this is unnecessary, and may be detrimental. All ADS domains will automatically create SRV records in the DNS zone `_kerberos.REALM.NAME` for each KDC in the realm. MIT's, as well as Heimdal's, KRB5 libraries default to checking for these records, so they will automatically find the KDCs. In addition, `krb5.conf` only allows specifying a single KDC, even there if there is more than one. Using the DNS lookup allows the KRB5 libraries to use whichever KDCs are available.

When manually configuring `krb5.conf`, the minimal configuration is:

```
[libdefaults]
    default_realm = YOUR.KERBEROS.REALM

[realms]
    YOUR.KERBEROS.REALM = {
        kdc = your.kerberos.server
    }

[domain_realms]
    .kerberos.server = YOUR.KERBEROS.REALM
```

When using Heimdal versions before 0.6 use the following configuration settings:

```
[libdefaults]
    default_realm      = YOUR.KERBEROS.REALM
    default_etypes     = des-cbc-crc des-cbc-md5
    default_etypes_des = des-cbc-crc des-cbc-md5

[realms]
    YOUR.KERBEROS.REALM = {
        kdc = your.kerberos.server
    }

[domain_realms]
    .kerberos.server = YOUR.KERBEROS.REALM
```

Test your config by doing a `kinit USERNAME@REALM` and making sure that your password is accepted by the Win2000 KDC.

With Heimdal versions earlier than 0.6.x you only can use newly created accounts in ADS or accounts that have had the password changed once after migration, or in case of `Administrator` after installation. At the moment, a Windows 2003 KDC can only be used with a Heimdal releases later than 0.6 (and no default etypes in `krb5.conf`). Unfortunately this whole area is still in a state of flux.

NOTE



The realm must be in uppercase or you will get “*Cannot find KDC for requested realm while getting initial credentials*” error (Kerberos is case-sensitive!).

NOTE



Time between the two servers must be synchronized. You will get a “*kinit(v5): Clock skew too great while getting initial credentials*” if the time difference is more than five minutes. Clock skew limits are configurable in the Kerberos protocols. The default setting is five minutes.

You also must ensure that you can do a reverse DNS lookup on the IP address of your KDC. Also, the name that this reverse lookup maps to must either be the NetBIOS name of the KDC (i.e., the hostname with no domain attached) or it can alternately be the NetBIOS name followed by the realm.

The easiest way to ensure you get this right is to add a `/etc/hosts` entry mapping the IP address of your KDC to its NetBIOS name. If you do not get this correct then you will get a local error when you try to join the realm.

If all you want is Kerberos support in `smbclient` then you can skip directly to Section 6.4.5 now. Section 6.4.3 and Section 6.4.4 are needed only if you want Kerberos support for `smbd` and `winbindd`.

6.4.3 Create the Computer Account

As a user who has write permission on the Samba private directory (usually `root`), run:

```
root# net ads join -U Administrator%password
```

When making a Windows client a member of an ADS domain within a complex organization, you may want to create the machine account within a particular organizational unit. Samba-3 permits this to be done using the following syntax:

```
root# kinit Administrator@your.kerberos.REALM
root# net ads join organizational_unit
```

For example, you may want to create the machine account in a container called “*Servers*” under the organizational directory “*Computers\BusinessUnit\Department*” like this:

```
root# net ads join "Computers\BusinessUnit\Department\Servers"
```

6.4.3.1 Possible Errors

ADS support not compiled in — Samba must be reconfigured (remove `config.cache`) and recompiled (make clean all install) after the Kerberos libraries and headers files are installed.

net ads join prompts for user name — You need to login to the domain using `kinit USERNAME@REALM`. `USERNAME` must be a user who has rights to add a machine to the domain.

Unsupported encryption/or checksum types — Make sure that the `/etc/krb5.conf` is correctly configured for the type and version of Kerberos installed on the system.

6.4.4 Testing Server Setup

If the join was successful, you will see a new computer account with the NetBIOS name of your Samba server in Active Directory (in the “*Computers*” folder under Users and Computers).

On a Windows 2000 client, try `net use * \\server\share`. You should be logged in with Kerberos without needing to know a password. If this fails then run `klist tickets`. Did you get a ticket for the server? Does it have an encryption type of DES-CBC-MD5?

NOTE



Samba can use both DES-CBC-MD5 encryption as well as ARCFOUR-HMAC-MD5 encoding.

6.4.5 Testing with smbclient

On your Samba server try to login to a Win2000 server or your Samba server using `smbclient` and Kerberos. Use `smbclient` as usual, but specify the `-k` option to choose Kerberos authentication.

6.4.6 Notes

You must change administrator password at least once after DC install, to create the right encryption types.

Windows 200x does not seem to create the `_kerberos._udp` and `_ldap._tcp` in the default DNS setup. Perhaps this will be fixed later in service packs.

6.5 Sharing User ID Mappings between Samba Domain Members

Samba maps UNIX users and groups (identified by UIDs and GIDs) to Windows users and groups (identified by SIDs). These mappings are done by the `idmap` subsystem of Samba.

In some cases it is useful to share these mappings between Samba Domain Members, so `name->id` mapping is identical on all machines. This may be needed in particular when sharing files over both CIFS and NFS.

To use the LDAP `ldap idmap suffix`, set:

```
ldap idmap suffix = ou=Idmap,dc=quencya,dc=org
```

See the `smb.conf` man page entry for the `ldap idmap suffix` parameter for further information.

Do not forget to specify also the `ldap admin dn` and to make certain to set the LDAP administrative password into the `secrets.tdb` using:

```
root# smbpasswd -w ldap-admin-password
```

6.6 Common Errors

In the process of adding/deleting/re-adding Domain Member machine accounts, there are many traps for the unwary player and many “*little*” things that can go wrong. It is particularly interesting how often subscribers on the Samba mailing list have concluded after repeated failed attempts to add a machine account that it is necessary to “*re-install*” MS Windows on the machine. In truth, it is seldom necessary to reinstall because of this type of problem. The real solution is often quite simple and with an understanding of how MS Windows networking functions, it is easy to overcome.

6.6.1 Cannot Add Machine Back to Domain

“A Windows workstation was re-installed. The original domain machine account was deleted and added immediately. The workstation will not join the domain if I use the same machine name. Attempts to add the machine fail with a message that the machine already exists on the network — I know it does not. Why is this failing?”

The original name is still in the NetBIOS name cache and must expire after machine account deletion before adding that same name as a Domain Member again. The best advice is to delete the old account and then add the machine with a new name.

6.6.2 Adding Machine to Domain Fails

“Adding a Windows 200x or XP Professional machine to the Samba PDC Domain fails with a message that, ‘The machine could not be added at this time, there is a network problem. Please try again later.’ Why?”

You should check that there is an *add machine script* in your `smb.conf` file. If there is not, please add one that is appropriate for your OS platform. If a script has been defined, you will need to debug its operation. Increase the *log level* in the `smb.conf` file to level 10, then try to rejoin the domain. Check the logs to see which operation is failing.

Possible causes include:

- The script does not actually exist, or could not be located in the path specified.

Corrective action: Fix it. Make sure when run manually that the script will add both the UNIX system account and the Samba SAM account.

- The machine could not be added to the UNIX system accounts file `/etc/passwd`.

Corrective action: Check that the machine name is a legal UNIX system account name. If the UNIX utility `useradd` is called, then make sure that the machine name you are trying to add can be added using this tool. `Useradd` on some systems will not allow any upper case characters nor will it allow spaces in the name.

The *add machine script* does not create the machine account in the Samba backend database, it is there only to create a UNIX system account to which the Samba backend database account can be mapped.

6.6.3 I Can't Join a Windows 2003 PDC

Windows 2003 requires SMB signing. Client side SMB signing has been implemented in Samba-3.0. Set *client use spnego* = yes when communicating with a Windows 2003 server.

STAND-ALONE SERVERS

Stand-alone Servers are independent of Domain Controllers on the network. They are not Domain Members and function more like workgroup servers. In many cases a Stand-alone Server is configured with a minimum of security control with the intent that all data served will be readily accessible to all users.

7.1 Features and Benefits

Stand-alone Servers can be as secure or as insecure as needs dictate. They can have simple or complex configurations. Above all, despite the hoopla about Domain Security they remain a common installation.

If all that is needed is a server for read-only files, or for printers alone, it may not make sense to effect a complex installation. For example: A drafting office needs to store old drawings and reference standards. No one can write files to the server as it is legislatively important that all documents remain unaltered. A share mode read-only Stand-alone Server is an ideal solution.

Another situation that warrants simplicity is an office that has many printers that are queued off a single central server. Everyone needs to be able to print to the printers, there is no need to effect any access controls and no files will be served from the print server. Again, a share mode Stand-alone Server makes a great solution.

7.2 Background

The term *Stand-alone Server* means that it will provide local authentication and access control for all resources that are available from it. In general this means that there will be a local user database. In more technical terms, it means resources on the machine will be made available in either SHARE mode or in USER mode.

No special action is needed other than to create user accounts. Stand-alone servers do not provide network logon services. This means that machines that use this server do not perform a domain logon to it. Whatever logon facility the workstations are subject to is independent of this machine. It is, however, necessary to accommodate any network user

so the logon name they use will be translated (mapped) locally on the Stand-alone Server to a locally known user name. There are several ways this can be done.

Samba tends to blur the distinction a little in respect of what is a Stand-alone Server. This is because the authentication database may be local or on a remote server, even if from the SMB protocol perspective the Samba server is not a member of a domain security context.

Through the use of Pluggable Authentication Modules (PAM) and the name service switcher (NSSWITCH), which maintains the UNIX-user database) the source of authentication may reside on another server. We would be inclined to call this the authentication server. This means that the Samba server may use the local UNIX/Linux system password database (*/etc/passwd* or */etc/shadow*), may use a local *smbpasswd* file, or may use an LDAP backend, or even via PAM and Winbind another CIFS/SMB server for authentication.

7.3 Example Configuration

The examples, Example 7.1, and link `linkend="SimplePrintServer"/>`, are designed to inspire simplicity. It is too easy to attempt a high level of creativity and to introduce too much complexity in server and network design.

7.3.1 Reference Documentation Server

Configuration of a read-only data server that everyone can access is very simple. Example 7.1 is the `smb.conf` file that will do this. Assume that all the reference documents are stored in the directory `/export`, and the documents are owned by a user other than nobody. No home directories are shared, and there are no users in the `/etc/passwd` UNIX system database. This is a simple system to administer.

Example 7.1. `smb.conf` for Reference Documentation Server

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = GANDALF
security = SHARE
passdb backend = guest
wins server = 192.168.1.1

[data]
comment = Data
path = /export
guest only = Yes
```

In Example 7.1 above, the machine name is set to GANDALF, the workgroup is set to the name of the local workgroup (MIDEARTH) so the machine will appear together with systems with which users are familiar. The only password backend required is the “*guest*”

backend to allow default unprivileged account names to be used. As there is a WINS server on this network, we obviously make use of it.

7.3.2 Central Print Serving

Configuration of a simple print server is easy if you have all the right tools on your system.

ASSUMPTIONS:

1. The print server must require no administration.
2. The print spooling and processing system on our print server will be CUPS. (Please refer to Chapter 18, *CUPS Printing Support* for more information).
3. The print server will service only network printers. The network administrator will correctly configure the CUPS environment to support the printers.
4. All workstations will use only postscript drivers. The printer driver of choice is the one shipped with the Windows OS for the Apple Color LaserWriter.

In this example our print server will spool all incoming print jobs to `/var/spool/samba` until the job is ready to be submitted by Samba to the CUPS print processor. Since all incoming connections will be as the anonymous (guest) user, two things will be required:

ENABLING ANONYMOUS PRINTING

- The UNIX/Linux system must have a **guest** account. The default for this is usually the account **nobody**. To find the correct name to use for your version of Samba, do the following:

```
$ testparm -s -v | grep "guest account"
```

Make sure that this account exists in your system password database (`/etc/passwd`).

- The directory into which Samba will spool the file must have write access for the guest account. The following commands will ensure that this directory is available for use:

```
root# mkdir /var/spool/samba
root# chown nobody.nobody /var/spool/samba
root# chmod a+rwt /var/spool/samba
```

The contents of the `smb.conf` file is shown in Example 7.2.

Example 7.2. smb.conf for Anonymous Printing

```
# Global parameters

[global]
workgroup = MIDEARTH
netbios name = GANDALF
security = SHARE
passdb backend = guest
printing = cups
printcap name = cups

[printers]
comment = All Printers
path = /var/spool/samba
printer admin = root
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = No
```

NOTE

On CUPS-enabled systems there is a facility to pass raw data directly to the printer without intermediate processing via CUPS print filters. Where use of this mode of operation is desired, it is necessary to configure a raw printing device. It is also necessary to enable the raw mime handler in the `/etc/mime.conv` and `/etc/mime.types` files. Refer to Section 18.3.4.

7.4 Common Errors

The greatest mistake so often made is to make a network configuration too complex. It pays to use the simplest solution that will meet the needs of the moment.

MS WINDOWS NETWORK CONFIGURATION GUIDE

8.1 Features and Benefits

Occasionally network administrators will report difficulty getting Microsoft Windows clients to interoperate correctly with Samba servers. It would appear that some folks just can not accept the fact that the right way to configure MS Windows network client is precisely as one would do when using Microsoft Windows NT4 or 200x servers. Yet there is repetitious need to provide detailed Windows client configuration instructions.

The purpose of this chapter is to graphically illustrate MS Windows client configuration for the most common critical aspects of such configuration. An experienced network administrator will not be interested in the details of this chapter.

8.2 Technical Details

This chapter discusses TCP/IP protocol configuration as well as network membership for the platforms that are in common use today. These are:

- Microsoft Windows XP Professional.
- Windows 2000 Professional.
- Windows Millenium edition (Me).

8.2.1 TCP/IP Configuration

The builder of a house must ensure that all construction takes place on a firm foundation. The same is true of TCP/IP-based networking. Fundamental network configuration problems will plague all network users until they are resolved.

Microsoft Windows workstations and servers can be configured either with fixed IP addresses or via DHCP. The examples that follow demonstrate the use of DHCP and make only passing reference to those situations where fixed IP configuration settings can be effected.

It is possible to use shortcuts or abbreviated keystrokes to arrive at a particular configuration screen. The decision was made to base all examples in this chapter on use of the **Start** button.

8.2.1.1 MS Windows XP Professional

There are two paths to the Windows XP TCP/IP configuration panel. Choose the access method that you prefer:

Click **Start** -> **Control Panel** -> **Network Connections**

Alternately, click **Start** ->, and right click **My Network Places** then select **Properties**

The following procedure steps through the Windows XP Professional TCP/IP configuration process:

1. On some installations the interface will be called **Local Area Connection** and on others it will be called **Network Bridge**. On our system it is called **Network Bridge**. Right click on **Network Bridge** -> **Properties**. See Figure 8.1.

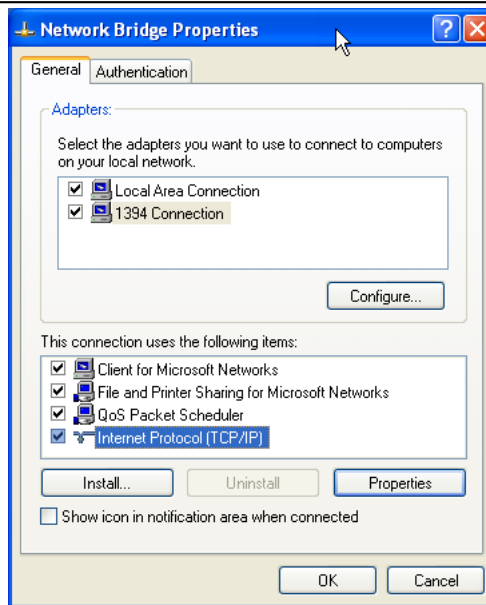


Figure 8.1. Network Bridge Configuration.

2. The Network Bridge Configuration, or Local Area Connection, panel is used to set TCP/IP protocol settings. In **This connection uses the following items:** box, click on **Internet Protocol (TCP/IP)**, then click the on **Properties**. The default setting is DHCP enabled operation. (i.e., “*Obtain an IP address automatically*”). See Figure 8.2.

Many network administrators will want to use DHCP to configure all client TCP/IP protocol stack settings. (For information on how to configure the ISC DHCP server for Microsoft Windows client support see, Section 39.2.2. If it is necessary to provide

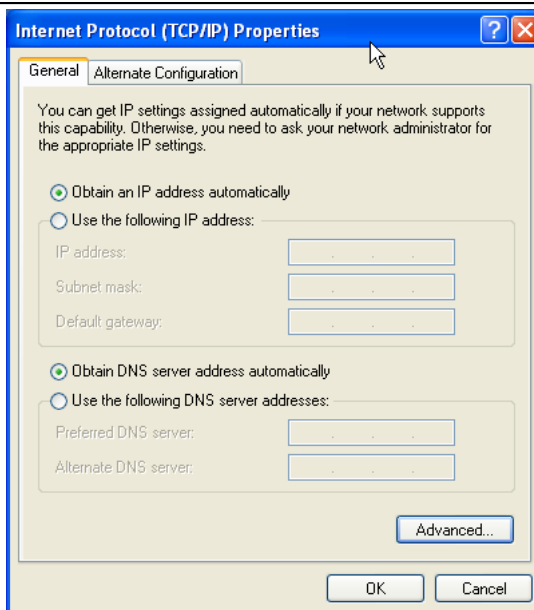


Figure 8.2. Internet Protocol (TCP/IP) Properties.

a fixed IP address, click on “*Use the following IP address*” and proceed to enter the IP Address, the subnet mask, and the default gateway address in the boxes provided.

3. Click the **Advanced** button to proceed with TCP/IP configuration. This opens a panel in which it is possible to create additional IP Addresses for this interface. The technical name for the additional addresses is *IP Aliases*, and additionally this panel permits the setting of more default gateways (routers). In most cases where DHCP is used, it will not be necessary to create additional settings. See Figure 8.3 to see the appearance of this panel.

Fixed settings may be required for DNS and WINS if these settings are not provided automatically via DHCP.

4. Click the **DNS** tab to add DNS server settings. The example system uses manually configured DNS settings. When finished making changes, click the **OK** to commit the settings. See Figure 8.4.
5. Click the **WINS** tab to add manual WINS server entries. This step demonstrates an example system that uses manually configured WINS settings. When finished making changes click the **OK** to commit the settings. See Figure 8.5.

8.2.1.2 MS Windows 2000

There are two paths to the Windows 2000 Professional TCP/IP configuration panel. Choose the access method that you prefer:

Click **Start -> Control Panel -> Network and Dial-up Connections**

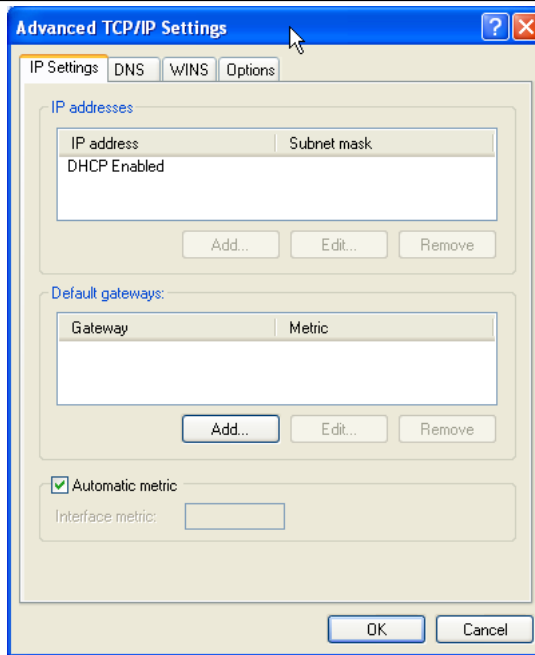


Figure 8.3. Advanced Network Settings

Alternately, click on **Start**, then right click **My Network Places** and select **Properties**.

The following procedure steps through the Windows XP Professional TCP/IP configuration process:

1. Right click on **Local Area Connection**, now click the **Properties**. See Figure 8.6.
2. The Local Area Connection Properties is used to set TCP/IP protocol settings. Click on **Internet Protocol (TCP/IP)** in the **Components checked are used by this connection:** box, then click the **Properties** button.
3. The default setting is DHCP enabled operation. (i.e., “*Obtain an IP address automatically*”). See Figure 8.7.

Many network administrators will want to use DHCP to configure all client TCP/IP protocol stack settings. (For information on how to configure the ISC DHCP server for Microsoft Windows client support, see Section 39.2.2. If it is necessary to provide a fixed IP address, click on “*Use the following IP address*” and proceed to enter the IP Address, the subnet mask, and the default gateway address in the boxes provided. For this example we are assuming that all network clients will be configured using DHCP.

4. Click the **Advanced** button to proceed with TCP/IP configuration. Refer to Figure 8.8.

Fixed settings may be required for DNS and WINS if these settings are not provided automatically via DHCP.

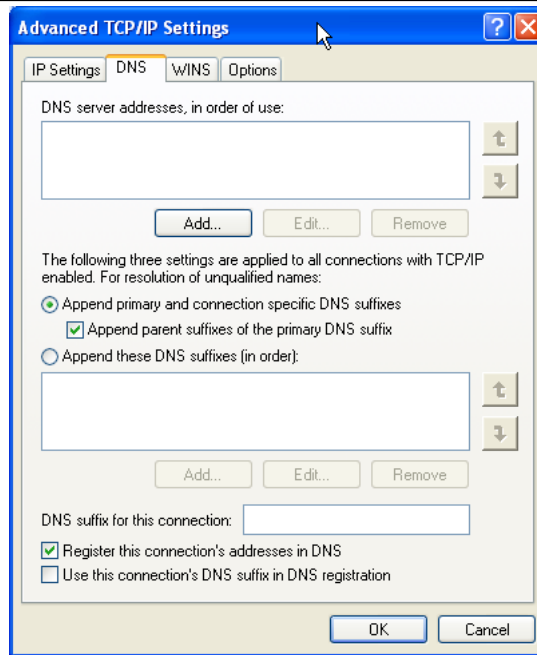


Figure 8.4. DNS Configuration.

5. Click the **DNS** tab to add DNS server settings. The example system uses manually configured DNS settings. When finished making changes, click on **OK** to commit the settings. See Figure 8.9.
6. Click the **WINS** tab to add manual WINS server entries. This step demonstrates an example system that uses manually configured WINS settings. When finished making changes, click on **OK** to commit the settings. See Figure 8.10.

8.2.1.3 MS Windows Me

There are two paths to the Windows Millenium edition (Me) TCP/IP configuration panel. Choose the access method that you prefer:

Click **Start -> Control Panel -> Network Connections**

Alternately, click on **Start ->**, and right click on **My Network Places** then select **Properties**.

The following procedure steps through the Windows Me TCP/IP configuration process:

1. In the box labelled **The following network components are installed:**, click on **Internet Protocol TCP/IP**, now click on the **Properties** button. See Figure 8.11.
2. Many network administrators will want to use DHCP to configure all client TCP/IP protocol stack settings. (For information on how to configure the ISC DHCP server for Microsoft Windows client support see, Section 39.2.2. The default setting on

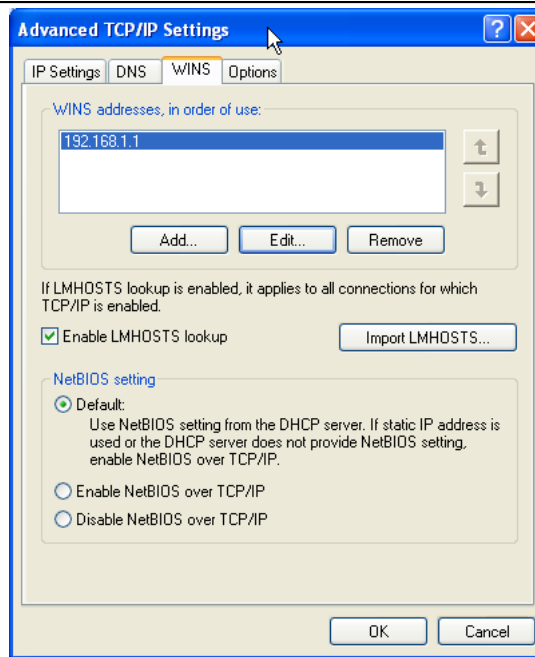


Figure 8.5. WINS Configuration

Microsoft Windows Me workstations is for DHCP enabled operation, i.e., **Obtain IP address automatically** is enabled. See Figure 8.12.

If it is necessary to provide a fixed IP address, click on **Specify an IP address** and proceed to enter the IP Address and the subnet mask in the boxes provided. For this example we are assuming that all network clients will be configured using DHCP.

3. Fixed settings may be required for DNS and WINS if these settings are not provided automatically via DHCP.
4. If necessary, click the **DNS Configuration** tab to add DNS server settings. Click the **WINS Configuration** tab to add WINS server settings. The **Gateway** tab allows additional gateways (router addresses) to be added to the network interface settings. In most cases where DHCP is used, it will not be necessary to create these manual settings.
5. The following example uses manually configured WINS settings. See Figure 8.13. When finished making changes, click on **OK** to commit the settings.

This is an example of a system that uses manually configured WINS settings. One situation where this might apply is on a network that has a single DHCP server that provides settings for multiple Windows workgroups or domains. See Figure 8.14.

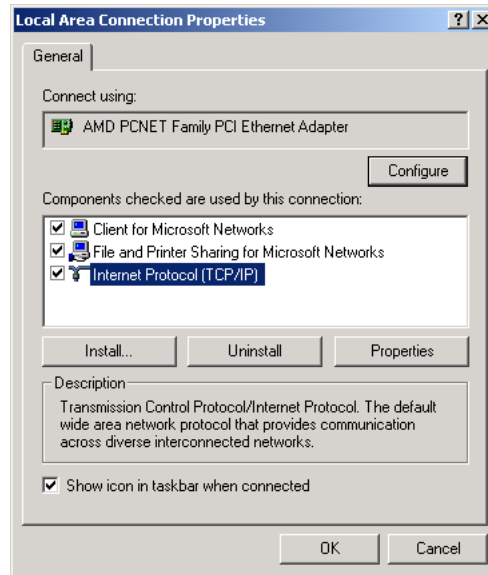


Figure 8.6. Local Area Connection Properties.

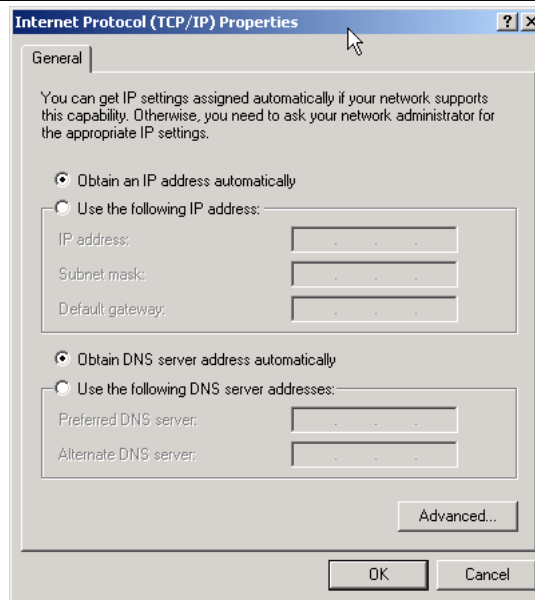


Figure 8.7. Internet Protocol (TCP/IP) Properties.

8.2.2 Joining a Domain: Windows 2000/XP Professional

Microsoft Windows NT/200x/XP Professional platforms can participate in Domain Security. This section steps through the process for making a Windows 200x/XP Professional machine

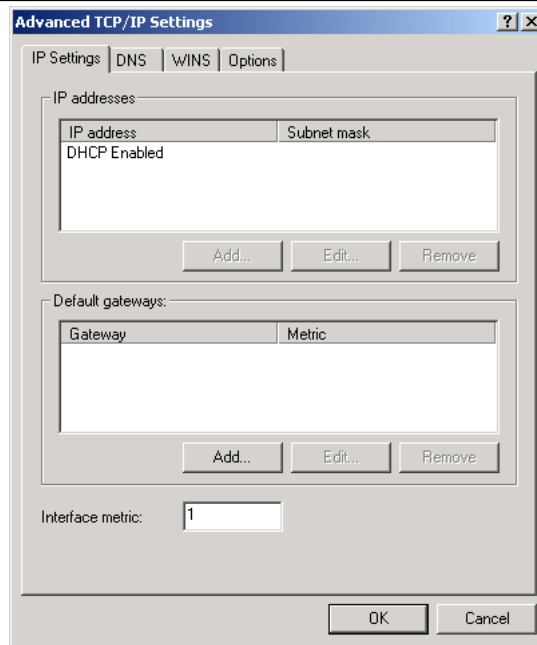


Figure 8.8. Advanced Network Settings.

a member of a Domain Security environment. It should be noted that this process is identical when joining a domain that is controlled by Windows NT4/200x as well as a Samba PDC.

1. Click **Start**.
2. Right click **My Computer**, then select **Properties**.
3. The opening panel is the same one that can be reached by clicking **System** on the Control Panel. See Figure 8.15.
4. Click the **Computer Name** tab. This panel shows the **Computer Description**, the **Full computer name**, and the **Workgroup** or **Domain name**. Clicking the **Network ID** button will launch the configuration wizard. Do not use this with Samba-3. If you wish to change the computer name, join or leave the domain, click the **Change** button. See Figure 8.16.
5. Click on **Change**. This panel shows that our example machine (TEMPTATION) is in a workgroup called WORKGROUP. We will join the domain called MIDEARTH. See Figure 8.17.
6. Enter the name **MIDEARTH** in the field below the Domain radio button. This panel shows that our example machine (TEMPTATION) is set to join the domain called MIDEARTH. See Figure 8.18.
7. Now click the **OK** button. A dialog box should appear to allow you to provide the credentials (username and password) of a Domain administrative account that has the rights to add machines to the Domain. Enter the name “*root*” and the root password

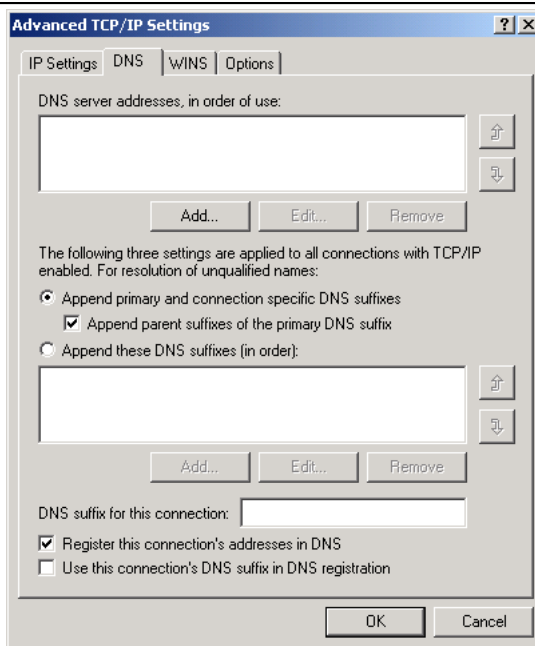


Figure 8.9. DNS Configuration.

from your Samba-3 server. See Figure 8.19.

8. Click on **OK**. The “*Welcome to the MIDEARTH domain.*” dialog box should appear. At this point the machine must be rebooted. Joining the domain is now complete.

8.2.3 Domain Logon Configuration: Windows 9x/Me

We follow the convention used by most in saying that Windows 9x/Me machines can participate in Domain logons. The truth is that these platforms can use only the LanManager network logon protocols.

NOTE



Windows XP Home edition cannot participate in Domain or LanManager network logons.

1. Right click on the **Network Neighborhood** icon.
2. The Network Configuration Panel allows all common network settings to be changed. See Figure 8.20.

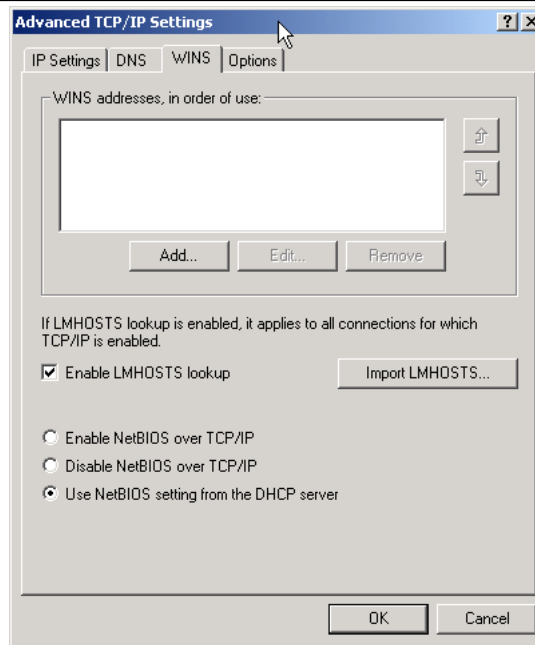


Figure 8.10. WINS Configuration.

Make sure that the **Client for Microsoft Networks** driver is installed as shown. Click on the **Client for Microsoft Networks** entry in **The following network components are installed:** box. Then click the **Properties** button.

3. The Client for Microsoft Networks Properties panel is the correct location to configure network logon settings. See Figure 8.21.

Enter the Windows NT domain name, check the **Log on to Windows NT domain** box, click **OK**.

4. Click on the **Identification** button. This is the location at which the workgroup (domain) name and the machine name (computer name) need to be set. See Figure 8.22.
5. Now click the **Access Control** button. If you want to be able to assign share access permissions using domain user and group accounts, it is necessary to enable **User-level access control** as shown in this panel. See Figure 8.23.

8.3 Common Errors

The most common errors that can afflict Windows networking systems include:

- Incorrect IP address.
- Incorrect or inconsistent netmasks.

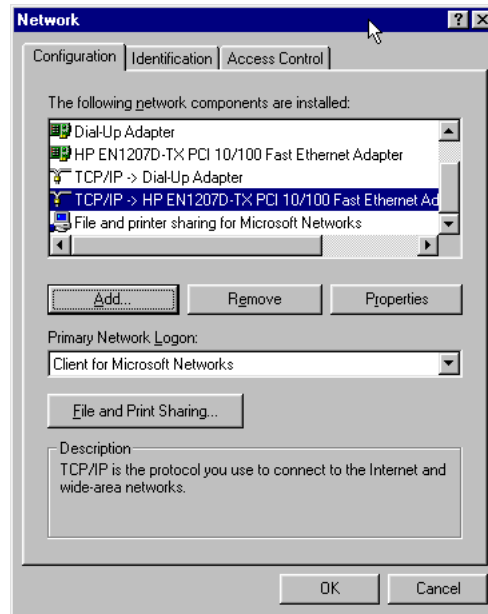


Figure 8.11. The Windows Me Network Configuration Panel.

- Incorrect router address.
- Incorrect DNS server address.
- Incorrect WINS server address.
- Use of a Network Scope setting — watch out for this one!

The most common reasons for which a Windows NT/200x/XP Professional client cannot join the Samba controlled domain are:

- `smb.conf` does not have correct *add machine script* settings.
- “*root*” account is not in password backend database.
- Attempt to use a user account instead of the “*root*” account to join a machine to the domain.
- Open connections from the workstation to the server.
- Firewall or filter configurations in place on either the client or on the Samba server.

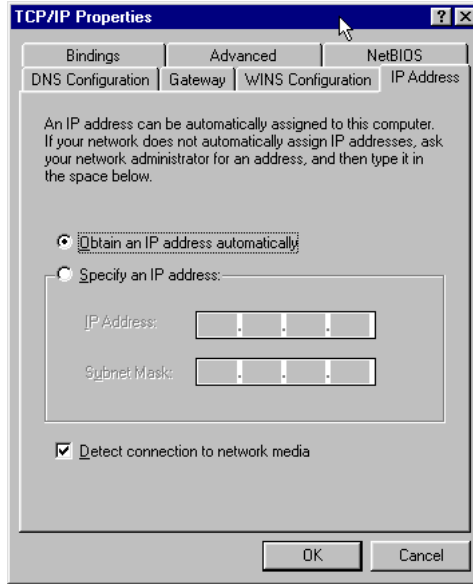


Figure 8.12. IP Address.

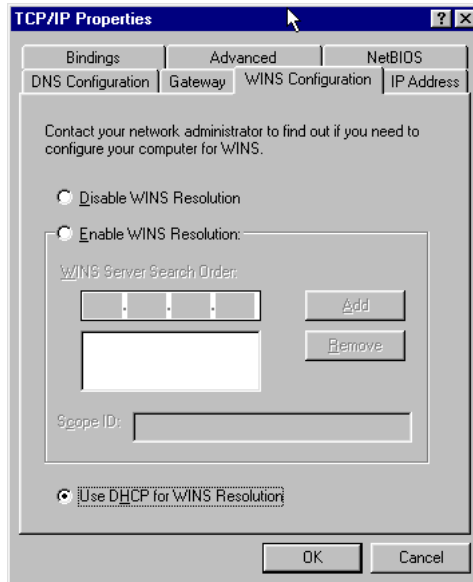


Figure 8.13. DNS Configuration.

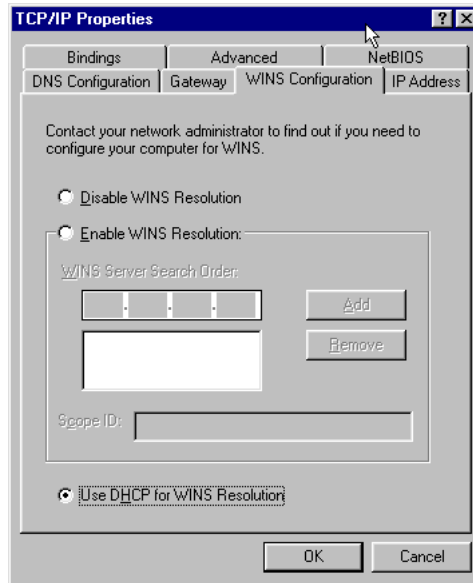


Figure 8.14. WINS Configuration.

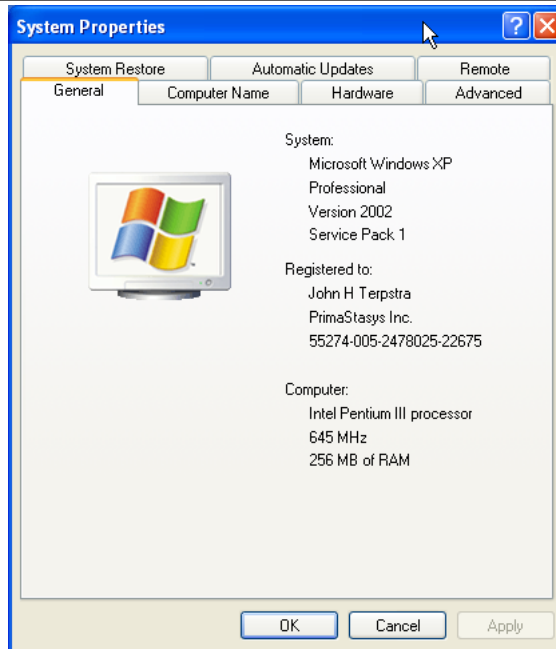


Figure 8.15. The General Panel.

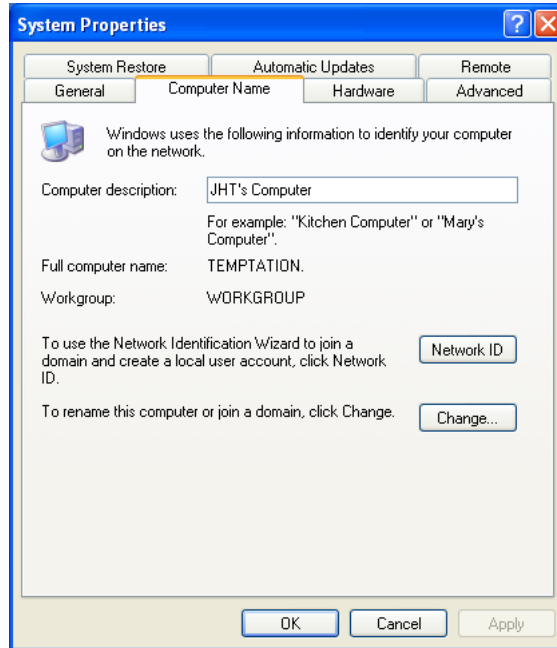


Figure 8.16. The Computer Name Panel.

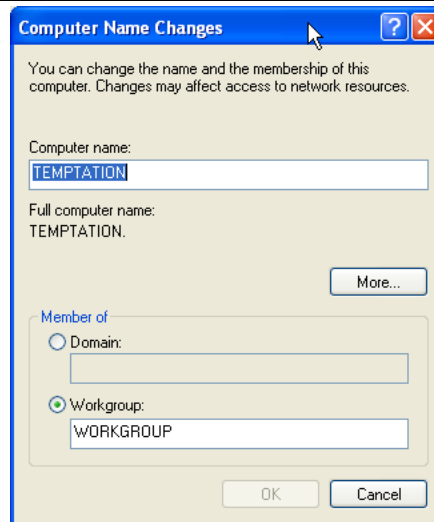


Figure 8.17. The Computer Name Changes Panel.

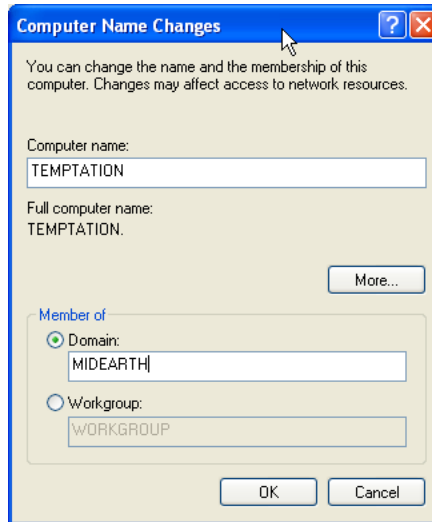


Figure 8.18. The Computer Name Changes Panel Domain MIDEARTH.



Figure 8.19. Computer Name Changes User name and Password Panel.

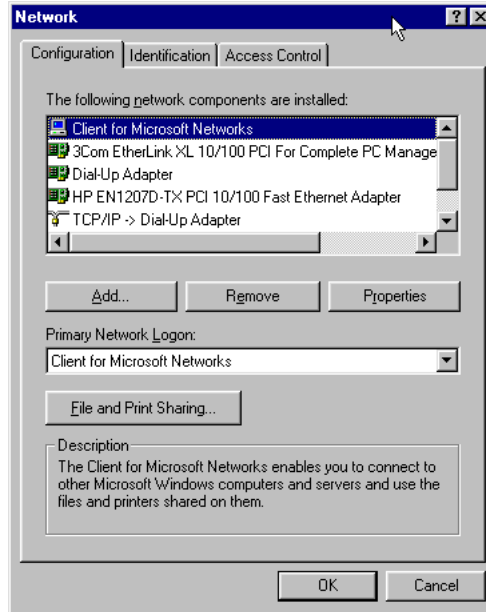


Figure 8.20. The Network Panel.

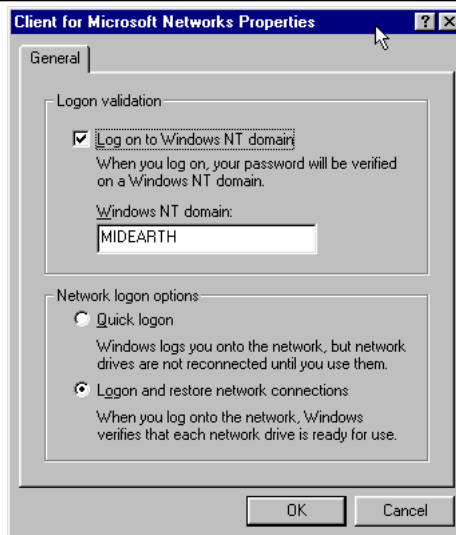


Figure 8.21. Client for Microsoft Networks Properties Panel.

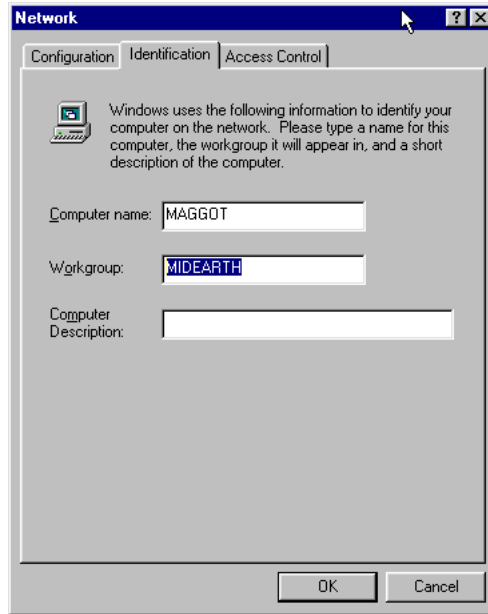


Figure 8.22. Identification Panel.

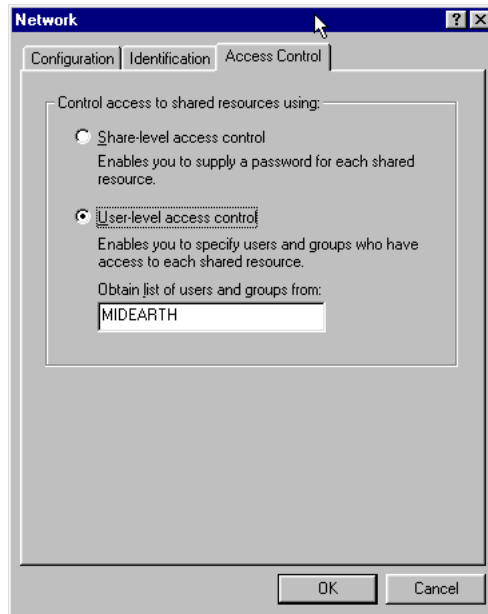


Figure 8.23. Identification Panel.

Part III

Advanced Configuration

NETWORK BROWSING

This document contains detailed information as well as a fast track guide to implementing browsing across subnets and/or across workgroups (or domains). WINS is the best tool for resolution of NetBIOS names to IP addresses. WINS is not involved in browse list handling except by way of name to address resolution.

NOTE



MS Windows 2000 and later versions can be configured to operate with no NetBIOS over TCP/IP. Samba-3 and later versions also support this mode of operation. When the use of NetBIOS over TCP/IP has been disabled, the primary means for resolution of MS Windows machine names is via DNS and Active Directory. The following information assumes that your site is running NetBIOS over TCP/IP.

9.1 Features and Benefits

Someone once referred to the past in these words “It was the best of times, it was the worst of times.” The more we look back, the more we long for what was and hope it never returns.

For many MS Windows network administrators, that statement sums up their feelings about NetBIOS networking precisely. For those who mastered NetBIOS networking, its fickle nature was just par for the course. For those who never quite managed to tame its lusty features, NetBIOS is like Paterson’s Curse.

For those not familiar with botanical problems in Australia, Paterson’s Curse, *Echium plantagineum*, was introduced to Australia from Europe during the mid-nineteenth century. Since then it has spread rapidly. The high seed production, with densities of thousands of seeds per square meter, a seed longevity of more than seven years, and an ability to germinate at any time of year, given the right conditions, are some of the features which make it such a persistent weed.

In this chapter we explore vital aspects of Server Message Block (SMB) networking with a particular focus on SMB as implemented through running NetBIOS (Network Basic In-

put/Output System) over TCP/IP. Since Samba does not implement SMB or NetBIOS over any other protocols, we need to know how to configure our network environment and simply remember to use nothing but TCP/IP on all our MS Windows network clients.

Samba provides the ability to implement a WINS (Windows Internetworking Name Server) and implements extensions to Microsoft's implementation of WINS. These extensions help Samba to effect stable WINS operations beyond the normal scope of MS WINS.

WINS is exclusively a service that applies only to those systems that run NetBIOS over TCP/IP. MS Windows 200x/XP have the capacity to operate with support for NetBIOS disabled, in which case WINS is of no relevance. Samba supports this also.

For those networks on which NetBIOS has been disabled (i.e., WINS is not required) the use of DNS is necessary for host name resolution.

9.2 What Is Browsing?

To most people browsing means they can see the MS Windows and Samba servers in the Network Neighborhood, and when the computer icon for a particular server is clicked, it opens up and shows the shares and printers available on the target server.

What seems so simple is in fact a complex interaction of different technologies. The technologies (or methods) employed in making all of this work include:

- MS Windows machines register their presence to the network.
- Machines announce themselves to other machines on the network.
- One or more machine on the network collates the local announcements.
- The client machine finds the machine that has the collated list of machines.
- The client machine is able to resolve the machine names to IP addresses.
- The client machine is able to connect to a target machine.

The Samba application that controls browse list management and name resolution is called `nmbd`. The configuration parameters involved in `nmbd`'s operation are:

Browsing options: *os level*(*), *lm announce*, *lm interval*, *preferred master*(*), *local master*(*), *domain master*(*), *browse list*, *enhanced browsing*.

Name Resolution Method: *name resolve order*(*).

WINS options: *dns proxy*, *wins proxy*, *wins server*(*), *wins support*(*), *wins hook*.

For Samba, the WINS Server and WINS Support are mutually exclusive options. Those marked with an (*) are the only options that commonly may need to be modified. Even if none of these parameters is set, `nmbd` will still do its job.

9.3 Discussion

All MS Windows networking uses SMB-based messaging. SMB messaging may be implemented with or without NetBIOS. MS Windows 200x supports NetBIOS over TCP/IP for

backwards compatibility. Microsoft appears intent on phasing out NetBIOS support.

9.3.1 NetBIOS over TCP/IP

Samba implements NetBIOS, as does MS Windows NT/200x/XP, by encapsulating it over TCP/IP. MS Windows products can do likewise. NetBIOS-based networking uses broadcast messaging to effect browse list management. When running NetBIOS over TCP/IP, this uses UDP-based messaging. UDP messages can be broadcast or unicast.

Normally, only unicast UDP messaging can be forwarded by routers. The *remote announce* parameter to `smb.conf` helps to project browse announcements to remote network segments via unicast UDP. Similarly, the *remote browse sync* parameter of `smb.conf` implements browse list collation using unicast UDP.

Secondly, in those networks where Samba is the only SMB server technology, wherever possible `nmbd` should be configured on one machine as the WINS server. This makes it easy to manage the browsing environment. If each network segment is configured with its own Samba WINS server, then the only way to get cross-segment browsing to work is by using the *remote announce* and the *remote browse sync* parameters to your `smb.conf` file.

If only one WINS server is used for an entire multi-segment network, then the use of the *remote announce* and the *remote browse sync* parameters should not be necessary.

As of Samba-3 WINS replication is being worked on. The bulk of the code has been committed, but it still needs maturation. This is not a supported feature of the Samba-3.0.0 release. Hopefully, this will become a supported feature of one of the Samba-3 release series.

Right now Samba WINS does not support MS-WINS replication. This means that when setting up Samba as a WINS server, there must only be one `nmbd` configured as a WINS server on the network. Some sites have used multiple Samba WINS servers for redundancy (one server per subnet) and then used *remote browse sync* and *remote announce* to effect browse list collation across all segments. Note that this means clients will only resolve local names, and must be configured to use DNS to resolve names on other subnets in order to resolve the IP addresses of the servers they can see on other subnets. This setup is not recommended, but is mentioned as a practical consideration (i.e., an “*if all else fails*” scenario).

Lastly, take note that browse lists are a collection of unreliable broadcast messages that are repeated at intervals of not more than 15 minutes. This means that it will take time to establish a browse list and it can take up to 45 minutes to stabilize, particularly across network segments.

9.3.2 TCP/IP without NetBIOS

All TCP/IP-enabled systems use various forms of host name resolution. The primary methods for TCP/IP hostname resolution involve either a static file (`/etc/hosts`) or the Domain Name System (DNS). DNS is the technology that makes the Internet usable. DNS-based host name resolution is supported by nearly all TCP/IP-enabled systems. Only a few embedded TCP/IP systems do not support DNS.

When an MS Windows 200x/XP system attempts to resolve a host name to an IP address it follows a defined path:

1. Checks the `hosts` file. It is located in `C:\Windows NT\System32\Drivers\etc`.
2. Does a DNS lookup.
3. Checks the NetBIOS name cache.
4. Queries the WINS server.
5. Does a broadcast name lookup over UDP.
6. Looks up entries in `LMHOSTS`, located in `C:\Windows NT\System32\Drivers\etc`.

Windows 200x/XP can register its host name with a Dynamic DNS server. You can force register with a Dynamic DNS server in Windows 200x/XP using: `ipconfig /registerdns`.

With Active Directory (ADS), a correctly functioning DNS server is absolutely essential. In the absence of a working DNS server that has been correctly configured, MS Windows clients and servers will be unable to locate each other, so consequently network services will be severely impaired.

The use of Dynamic DNS is highly recommended with Active Directory, in which case the use of BIND9 is preferred for its ability to adequately support the SRV (service) records that are needed for Active Directory.

9.3.3 DNS and Active Directory

Occasionally we hear from UNIX network administrators who want to use a UNIX-based Dynamic DNS server in place of the Microsoft DNS server. While this might be desirable to some, the MS Windows 200x DNS server is auto-configured to work with Active Directory. It is possible to use BIND version 8 or 9, but it will almost certainly be necessary to create service records so MS Active Directory clients can resolve host names to locate essential network services. The following are some of the default service records that Active Directory requires:

`_ldap._tcp.pdc.ms-dcs.Domain` — This provides the address of the Windows NT PDC for the Domain.

`_ldap._tcp.pdc.ms-dcs.DomainTree` — Resolves the addresses of Global Catalog servers in the domain.

`_ldap._tcp.site.sites.writable.ms-dcs.Domain` — Provides list of Domain Controllers based on sites.

`_ldap._tcp.writable.ms-dcs.Domain` — Enumerates list of Domain Controllers that have the writable copies of the Active Directory data store.

`_ldap._tcp.GUID.domains.ms-dcs.DomainTree` — Entry used by MS Windows clients to locate machines using the Global Unique Identifier.

`_ldap._tcp.Site.gc.ms-dcs.DomainTree` — Used by MS Windows clients to locate site configuration dependent Global Catalog server.

9.4 How Browsing Functions

MS Windows machines register their NetBIOS names (i.e., the machine name for each service type in operation) on start-up. The exact method by which this name registration takes place is determined by whether or not the MS Windows client/server has been given a WINS server address, whether or not LMHOSTS lookup is enabled, or if DNS for NetBIOS name resolution is enabled, etc.

In the case where there is no WINS server, all name registrations as well as name lookups are done by UDP broadcast. This isolates name resolution to the local subnet, unless LMHOSTS is used to list all names and IP addresses. In such situations, Samba provides a means by which the Samba server name may be forcibly injected into the browse list of a remote MS Windows network (using the *remote announce* parameter).

Where a WINS server is used, the MS Windows client will use UDP unicast to register with the WINS server. Such packets can be routed and thus WINS allows name resolution to function across routed networks.

During the startup process an election will take place to create a Local Master Browser if one does not already exist. On each NetBIOS network one machine will be elected to function as the Domain Master Browser. This domain browsing has nothing to do with MS security Domain Control. Instead, the Domain Master Browser serves the role of contacting each local master browser (found by asking WINS or from LMHOSTS) and exchanging browse list contents. This way every master browser will eventually obtain a complete list of all machines that are on the network. Every 11 to 15 minutes an election is held to determine which machine will be the master browser. By the nature of the election criteria used, the machine with the highest uptime, or the most senior protocol version or other criteria, will win the election as Domain Master Browser.

Clients wishing to browse the network make use of this list, but also depend on the availability of correct name resolution to the respective IP address/addresses.

Any configuration that breaks name resolution and/or browsing intrinsics will annoy users because they will have to put up with protracted inability to use the network services.

Samba supports a feature that allows forced synchronization of browse lists across routed networks using the *remote browse sync* parameter in the `smb.conf` file. This causes Samba to contact the local master browser on a remote network and to request browse list synchronization. This effectively bridges two networks that are separated by routers. The two remote networks may use either broadcast-based name resolution or WINS-based name resolution, but it should be noted that the *remote browse sync* parameter provides browse list synchronization — and that is distinct from name to address resolution. In other

words, for cross-subnet browsing to function correctly it is essential that a name-to-address resolution mechanism be provided. This mechanism could be via DNS, */etc/hosts*, and so on.

9.4.1 Configuring WORKGROUP Browsing

To configure cross-subnet browsing on a network containing machines in a WORKGROUP, not an NT Domain, you need to set up one Samba server to be the Domain Master Browser (note that this is not the same as a Primary Domain Controller, although in an NT Domain the same machine plays both roles). The role of a Domain Master Browser is to collate the browse lists from Local Master Browsers on all the subnets that have a machine participating in the workgroup. Without one machine configured as a Domain Master Browser, each subnet would be an isolated workgroup unable to see any machines on another subnet. It is the presence of a Domain Master Browser that makes cross-subnet browsing possible for a workgroup.

In a WORKGROUP environment the Domain Master Browser must be a Samba server, and there must only be one Domain Master Browser per workgroup name. To set up a Samba server as a Domain Master Browser, set the following option in the *[global]* section of the *smb.conf* file:

```
domain master = yes
```

The Domain Master Browser should preferably be the local master browser for its own subnet. In order to achieve this, set the following options in the *[global]* section of the *smb.conf* file as shown in Example 9.1.

Example 9.1. Domain Master Browser *smb.conf*

```
[global]  
domain master = yes  
local master = yes  
preferred master = yes  
os level = 65
```

The Domain Master Browser may be the same machine as the WINS server, if necessary.

Next, you should ensure that each of the subnets contains a machine that can act as a Local Master Browser for the workgroup. Any MS Windows NT/200x/XP machine should be able to do this, as will Windows 9x/Me machines (although these tend to get rebooted more often, so it is not such a good idea to use these). To make a Samba server a Local Master Browser set the following options in the *[global]* section of the *smb.conf* file as shown in Example 9.2:

Do not do this for more than one Samba server on each subnet, or they will war with each other over which is to be the Local Master Browser.

The *local master* parameter allows Samba to act as a Local Master Browser. The *preferred master* causes **nmbd** to force a browser election on startup and the *os level* parameter sets Samba high enough so it should win any browser elections.

Example 9.2. Local master browser smb.conf

```
[global]
domain master = no
local master = yes
preferred master = yes
os level = 65
```

If you have an NT machine on the subnet that you wish to be the Local Master Browser, you can disable Samba from becoming a Local Master Browser by setting the following options in the `[global]` section of the `smb.conf` file as shown in Example 9.3:

Example 9.3. smb.conf for not being a Master Browser

```
[global]
domain master = no
local master = no
preferred master = no
os level = 0
```

9.4.2 DOMAIN Browsing Configuration

If you are adding Samba servers to a Windows NT Domain, then you must not set up a Samba server as a Domain Master Browser. By default, a Windows NT Primary Domain Controller for a domain is also the Domain Master Browser for that domain. Network browsing may break if a Samba server registers the domain master browser NetBIOS name (`DOMAIN<1B>`) with WINS instead of the PDC.

For subnets other than the one containing the Windows NT PDC, you may set up Samba servers as Local Master Browsers as described. To make a Samba server a Local Master Browser, set the following options in the `[global]` section of the `smb.conf` file as shown in Example 9.4:

Example 9.4. Local Master Browser smb.conf

```
[global]
domain master = no
local master = yes
preferred master = yes
os level = 65
```

If you wish to have a Samba server fight the election with machines on the same subnet you may set the `os level` parameter to lower levels. By doing this you can tune the order of

machines that will become Local Master Browsers if they are running. For more details on this refer to Section 9.4.3.

If you have Windows NT machines that are members of the domain on all subnets and you are sure they will always be running, you can disable Samba from taking part in browser elections and ever becoming a Local Master Browser by setting the following options in the *[global]* section of the `smb.conf` file as shown in Example 9.5:

Example 9.5. `smb.conf` for not being a master browser

```
[global]
domain master = no
local master = no
preferred master = no
os level = 0
```

9.4.3 Forcing Samba to Be the Master

Who becomes the master browser is determined by an election process using broadcasts. Each election packet contains a number of parameters that determine what precedence (bias) a host should have in the election. By default Samba uses a low precedence and thus loses elections to just about every Windows network server or client.

If you want Samba to win elections, set the *os level* global option in `smb.conf` to a higher number. It defaults to zero. Using 34 would make it win all elections every other system (except other samba systems).

An *os level* of two would make it beat Windows for Workgroups and Windows 9x/Me, but not MS Windows NT/200x Server. An MS Windows NT/200x Server Domain Controller uses level 32. The maximum os level is 255.

If you want Samba to force an election on startup, set the *preferred master* global option in `smb.conf` to `yes`. Samba will then have a slight advantage over other potential master browsers that are not Preferred Master Browsers. Use this parameter with care, as if you have two hosts (whether they are Windows 9x/Me or NT/200x/XP or Samba) on the same local subnet both set with *preferred master* to `yes`, then periodically and continually they will force an election in order to become the Local Master Browser.

If you want Samba to be a *Domain Master Browser*, then it is recommended that you also set *preferred master* to `yes`, because Samba will not become a Domain Master Browser for the whole of your LAN or WAN if it is not also a Local Master Browser on its own broadcast isolated subnet.

It is possible to configure two Samba servers to attempt to become the Domain Master Browser for a domain. The first server that comes up will be the Domain Master Browser. All other Samba servers will attempt to become the Domain Master Browser every five minutes. They will find that another Samba server is already the domain master browser and will fail. This provides automatic redundancy, should the current Domain Master Browser fail.

9.4.4 Making Samba the Domain Master

The domain master is responsible for collating the browse lists of multiple subnets so browsing can occur between subnets. You can make Samba act as the Domain Master by setting *domain master* = yes in `smb.conf`. By default it will not be a Domain Master.

Do not set Samba to be the Domain Master for a workgroup that has the same name as an NT/200x Domain. If Samba is configured to be the Domain Master for a workgroup that is present on the same network as a Windows NT/200x domain that has the same name, network browsing problems will certainly be experienced.

When Samba is the Domain Master and the Master Browser, it will listen for master announcements (made roughly every twelve minutes) from Local Master Browsers on other subnets and then contact them to synchronize browse lists.

If you want Samba to be the domain master, you should also set the *os level* high enough to make sure it wins elections, and set *preferred master* to *yes*, to get Samba to force an election on startup.

All servers (including Samba) and clients should be using a WINS server to resolve NetBIOS names. If your clients are only using broadcasting to resolve NetBIOS names, then two things will occur:

1. Local Master Browsers will be unable to find a Domain Master Browser, as they will be looking only on the local subnet.
2. If a client happens to get hold of a domain-wide browse list and a user attempts to access a host in that list, it will be unable to resolve the NetBIOS name of that host.

If, however, both Samba and your clients are using a WINS server, then:

1. Local master browsers will contact the WINS server and, as long as Samba has registered that it is a Domain Master Browser with the WINS server, the Local Master Browser will receive Samba's IP address as its Domain Master Browser.
2. When a client receives a domain-wide browse list and a user attempts to access a host in that list, it will contact the WINS server to resolve the NetBIOS name of that host. As long as that host has registered its NetBIOS name with the same WINS server, the user will be able to see that host.

9.4.5 Note about Broadcast Addresses

If your network uses a 0 based broadcast address (for example, if it ends in a 0) then you will strike problems. Windows for Workgroups does not seem to support a zeros broadcast and you will probably find that browsing and name lookups will not work.

9.4.6 Multiple Interfaces

Samba supports machines with multiple network interfaces. If you have multiple interfaces, you will need to use the *interfaces* option in `smb.conf` to configure them.

9.4.7 Use of the Remote Announce Parameter

The *remote announce* parameter of `smb.conf` can be used to forcibly ensure that all the NetBIOS names on a network get announced to a remote network. The syntax of the *remote announce* parameter is:

```
remote announce = a.b.c.d [e.f.g.h] ...
```

or

```
remote announce = a.b.c.d/WORKGROUP [e.f.g.h/WORKGROUP] ...
```

where:

a.b.c.d and *e.f.g.h* — is either the LMB (Local Master Browser) IP address or the broadcast address of the remote network. i.e., the LMB is at 192.168.1.10, or the address could be given as 192.168.1.255 where the netmask is assumed to be 24 bits (255.255.255.0). When the remote announcement is made to the broadcast address of the remote network, every host will receive our announcements. This is noisy and therefore undesirable but may be necessary if we do not know the IP address of the remote LMB.

WORKGROUP — is optional and can be either our own workgroup or that of the remote network. If you use the workgroup name of the remote network, our NetBIOS machine names will end up looking like they belong to that workgroup. This may cause name resolution problems and should be avoided.

9.4.8 Use of the Remote Browse Sync Parameter

The *remote browse sync* parameter of `smb.conf` is used to announce to another LMB that it must synchronize its NetBIOS name list with our Samba LMB. This works only if the Samba server that has this option is simultaneously the LMB on its network segment.

The syntax of the *remote browse sync* parameter is:

```
remote browse sync = a.b.c.d
```

where *a.b.c.d* is either the IP address of the remote LMB or else is the network broadcast address of the remote segment.

9.5 WINS — The Windows Internetworking Name Server

Use of WINS (either Samba WINS or MS Windows NT Server WINS) is highly recommended. Every NetBIOS machine registers its name together with a `name_type` value for each of several types of service it has available. It registers its name directly as a unique (the type 0x03) name. It also registers its name if it is running the LanManager compatible server service (used to make shares and printers available to other users) by registering the server (the type 0x20) name.

All NetBIOS names are up to 15 characters in length. The `name.type` variable is added to the end of the name, thus creating a 16 character name. Any name that is shorter than 15 characters is padded with spaces to the 15th character. Thus, all NetBIOS names are 16 characters long (including the `name.type` information).

WINS can store these 16-character names as they get registered. A client that wants to log onto the network can ask the WINS server for a list of all names that have registered the NetLogon service `name.type`. This saves broadcast traffic and greatly expedites logon processing. Since broadcast name resolution cannot be used across network segments this type of information can only be provided via WINS or via a statically configured `lmhosts` file that must reside on all clients in the absence of WINS.

WINS also serves the purpose of forcing browse list synchronization by all LMBs. LMBs must synchronize their browse list with the DMB (Domain Master Browser) and WINS helps the LMB to identify its DMB. By definition this will work only within a single workgroup. Note that the Domain Master Browser has nothing to do with what is referred to as an MS Windows NT Domain. The later is a reference to a security environment while the DMB refers to the master controller for browse list information only.

WINS will work correctly only if every client TCP/IP protocol stack has been configured to use the WINS servers. Any client that has not been configured to use the WINS server will continue to use only broadcast-based name registration so WINS may never get to know about it. In any case, machines that have not registered with a WINS server will fail name to address lookup attempts by other clients and will therefore cause workstation access errors.

To configure Samba as a WINS server just add `wins support = yes` to the `smb.conf` file `[global]` section.

To configure Samba to register with a WINS server just add `wins server = a.b.c.d` to your `smb.conf` file `[global]` section.

IMPORTANT



Never use both `wins support = yes` together with `wins server = a.b.c.d` particularly not using its own IP address. Specifying both will cause `nmbd` to refuse to start!

9.5.1 WINS Server Configuration

Either a Samba Server or a Windows NT Server machine may be set up as a WINS server. To configure a Samba Server to be a WINS server you must add to the `smb.conf` file on the selected Server the following line to the `[global]` section:

```
wins support = yes
```

Versions of Samba prior to 1.9.17 had this parameter default to `yes`. If you have any older versions of Samba on your network it is strongly suggested you upgrade to a recent version, or at the very least set the parameter to “`no`” on all these machines.

Machines configured with *wins support* = yes will keep a list of all NetBIOS names registered with them, acting as a DNS for NetBIOS names.

It is strongly recommended to set up only one WINS server. Do not set the *wins support* = yes option on more than one Samba server.

To configure Windows NT/200x Server as a WINS server, install and configure the WINS service. See the Windows NT/200x documentation for details. Windows NT/200x WINS servers can replicate to each other, allowing more than one to be set up in a complex subnet environment. As Microsoft refuses to document the replication protocols, Samba cannot currently participate in these replications. It is possible in the future that a Samba-to-Samba WINS replication protocol may be defined, in which case more than one Samba machine could be set up as a WINS server. Currently only one Samba server should have the *wins support* = yes parameter set.

After the WINS server has been configured, you must ensure that all machines participating on the network are configured with the address of this WINS server. If your WINS server is a Samba machine, fill in the Samba machine IP address in the **Primary WINS Server** field of the **Control Panel->Network->Protocols->TCP->WINS Server** dialogs in Windows 9x/Me or Windows NT/200x. To tell a Samba server the IP address of the WINS server, add the following line to the *[global]* section of all *smb.conf* files:

```
wins server = <name or IP address>
```

where <name or IP address> is either the DNS name of the WINS server machine or its IP address.

This line must not be set in the *smb.conf* file of the Samba server acting as the WINS server itself. If you set both the *wins support* = yes option and the *wins server* = <name> option then **nmbd** will fail to start.

There are two possible scenarios for setting up cross-subnet browsing. The first details setting up cross-subnet browsing on a network containing Windows 9x/Me, Samba and Windows NT/200x machines that are not configured as part of a Windows NT Domain. The second details setting up cross-subnet browsing on networks that contain NT Domains.

9.5.2 WINS Replication

Samba-3 permits WINS replication through the use of the *wrep1d* utility. This tool is not currently capable of being used as it is still in active development. As soon as this tool becomes moderately functional, we will prepare man pages and enhance this section of the documentation to provide usage and technical details.

9.5.3 Static WINS Entries

Adding static entries to your Samba WINS server is actually fairly easy. All you have to do is add a line to *wins.dat*, typically located in */usr/local/samba/var/locks*.

Entries in *wins.dat* take the form of:

```
"NAME#TYPE" TTL ADDRESS+ FLAGS
```

where NAME is the NetBIOS name, TYPE is the NetBIOS type, TTL is the time-to-live as an absolute time in seconds, ADDRESS+ is one or more addresses corresponding to the registration and FLAGS are the NetBIOS flags for the registration.

A typical dynamic entry looks like this:

```
"MADMAN#03" 1055298378 192.168.1.2 66R
```

To make it static, all that has to be done is set the TTL to 0, like this:

```
"MADMAN#03" 0 192.168.1.2 66R
```

Though this method works with early Samba-3 versions, there is a possibility that it may change in future versions if WINS replication is added.

9.6 Helpful Hints

The following hints should be carefully considered as they are stumbling points for many new network administrators.

9.6.1 Windows Networking Protocols

WARNING



Do not use more than one protocol on MS Windows machines.

A common cause of browsing problems results from installing more than one protocol on an MS Windows machine.

Every NetBIOS machine takes part in a process of electing the LMB (and DMB) every 15 minutes. A set of election criteria is used to determine the order of precedence for winning this election process. A machine running Samba or Windows NT will be biased so the most suitable machine will predictably win and thus retain its role.

The election process is “*fought out*” so to speak over every NetBIOS network interface. In the case of a Windows 9x/Me machine that has both TCP/IP and IPX installed and has NetBIOS enabled over both protocols, the election will be decided over both protocols. As often happens, if the Windows 9x/Me machine is the only one with both protocols then the LMB may be won on the NetBIOS interface over the IPX protocol. Samba will then lose

the LMB role as Windows 9x/Me will insist it knows who the LMB is. Samba will then cease to function as an LMB and thus browse list operation on all TCP/IP-only machines will fail.

Windows 95, 98, 98se, and Me are referred to generically as Windows 9x/Me. The Windows NT4, 200x, and XP use common protocols. These are roughly referred to as the Windows NT family, but it should be recognized that 2000 and XP/2003 introduce new protocol extensions that cause them to behave differently from MS Windows NT4. Generally, where a server does not support the newer or extended protocol, these will fall back to the NT4 protocols.

The safest rule of all to follow is: use only one protocol!

9.6.2 Name Resolution Order

Resolution of NetBIOS names to IP addresses can take place using a number of methods. The only ones that can provide NetBIOS `name_type` information are:

- WINS — the best tool.
- LMHOSTS — static and hard to maintain.
- Broadcast — uses UDP and cannot resolve names across remote segments.

Alternative means of name resolution include:

- Static `/etc/hosts`— hard to maintain, and lacks `name_type` info.
- DNS — is a good choice but lacks essential `name_type` info.

Many sites want to restrict DNS lookups and avoid broadcast name resolution traffic. The `name resolve order` parameter is of great help here. The syntax of the `name resolve order` parameter is:

```
name resolve order = wins lmhosts bcast host
```

or

```
name resolve order = wins lmhosts (eliminates bcast and host)
```

The default is:

```
name resolve order = host lmhost wins bcast
```

where “*host*” refers to the native methods used by the UNIX system to implement the `gethostbyname()` function call. This is normally controlled by `/etc/host.conf`, `/etc/nsswitch.conf` and `/etc/resolv.conf`.

9.7 Technical Overview of Browsing

SMB networking provides a mechanism by which clients can access a list of machines in a network, a so-called *browse list*. This list contains machines that are ready to offer file and/or print services to other machines within the network. Thus it does not include machines that aren't currently able to do server tasks. The browse list is heavily used by all

SMB clients. Configuration of SMB browsing has been problematic for some Samba users, hence this document.

MS Windows 2000 and later versions, as with Samba-3 and later versions, can be configured to not use NetBIOS over TCP/IP. When configured this way, it is imperative that name resolution (using DNS/LDAP/ADS) be correctly configured and operative. Browsing will not work if name resolution from SMB machine names to IP addresses does not function correctly.

Where NetBIOS over TCP/IP is enabled, use of a WINS server is highly recommended to aid the resolution of NetBIOS (SMB) names to IP addresses. WINS allows remote segment clients to obtain NetBIOS name_type information that cannot be provided by any other means of name resolution.

9.7.1 Browsing Support in Samba

Samba facilitates browsing. The browsing is supported by `nmbd` and is also controlled by options in the `smb.conf` file. Samba can act as a local browse master for a workgroup and the ability to support domain logons and scripts is now available.

Samba can also act as a Domain Master Browser for a workgroup. This means that it will collate lists from Local Master Browsers into a wide area network server list. In order for browse clients to resolve the names they may find in this list, it is recommended that both Samba and your clients use a WINS server.

Do not set Samba to be the Domain Master for a workgroup that has the same name as an NT Domain. On each wide area network, you must only ever have one Domain Master Browser per workgroup, regardless of whether it is NT, Samba or any other type of domain master that is providing this service.

NOTE



nmbd can be configured as a WINS server, but it is not necessary to specifically use Samba as your WINS server. MS Windows NT4, Server or Advanced Server 200x can be configured as your WINS server. In a mixed NT/200x server and Samba environment on a Wide Area Network, it is recommended that you use the Microsoft WINS server capabilities. In a Samba-only environment, it is recommended that you use one and only one Samba server as the WINS server.

To get browsing to work you need to run `nmbd` as usual, but will need to use the *workgroup* option in `smb.conf` to control what workgroup Samba becomes a part of.

Samba also has a useful option for a Samba server to offer itself for browsing on another subnet. It is recommended that this option is only used for “*unusual*” purposes: announcements over the Internet, for example. See *remote announce* in the `smb.conf` man page.

9.7.2 Problem Resolution

If something does not work, the `log.nmbd` file will help to track down the problem. Try a *log level* of 2 or 3 for finding problems. Also note that the current browse list usually gets stored in text form in a file called `browse.dat`.

If it does not work, you should still be able to type the server name as `\\SERVER` in **filemanager**, then press enter and **filemanager** should display the list of available shares.

Some people find browsing fails because they do not have the global *guest account* set to a valid account. Remember that the IPC\$ connection that lists the shares is done as guest and, thus, you must have a valid guest account.

MS Windows 2000 and later (as with Samba) can be configured to disallow anonymous (i.e., guest account) access to the IPC\$ share. In that case, the MS Windows 2000/XP/2003 machine acting as an SMB/CIFS client will use the name of the currently logged-in user to query the IPC\$ share. MS Windows 9x/Me clients are not able to do this and thus will not be able to browse server resources.

The other big problem people have is that their broadcast address, netmask or IP address is wrong (specified with the *interfaces* option in `smb.conf`)

9.7.3 Cross-Subnet Browsing

Since the release of Samba 1.9.17 (alpha1), Samba has supported the replication of browse lists across subnet boundaries. This section describes how to set this feature up in different settings.

To see browse lists that span TCP/IP subnets (i.e., networks separated by routers that do not pass broadcast traffic), you must set up at least one WINS server. The WINS server acts as a DNS for NetBIOS names. This will allow NetBIOS name-to-IP address translation to be completed by a direct query of the WINS server. This is done via a directed UDP packet on port 137 to the WINS server machine. The WINS server avoids the necessity of default NetBIOS name-to-IP address translation, which is done using UDP broadcasts from the querying machine. This means that machines on one subnet will not be able to resolve the names of machines on another subnet without using a WINS server.

Remember, for browsing across subnets to work correctly, all machines, be they Windows 95, Windows NT or Samba servers, must have the IP address of a WINS server given to them by a DHCP server, or by manual configuration (for Windows 9x/Me and Windows NT/200x/XP, this is in the TCP/IP Properties, under Network settings); for Samba, this is in the `smb.conf` file.

9.7.3.1 Behavior of Cross-Subnet Browsing

Cross-subnet Browsing is a complicated dance, containing multiple moving parts. It has taken Microsoft several years to get the code that achieves this correct, and Samba lags behind in some areas. Samba is capable of cross-subnet browsing when configured correctly.

Consider a network set up as Figure 9.1.

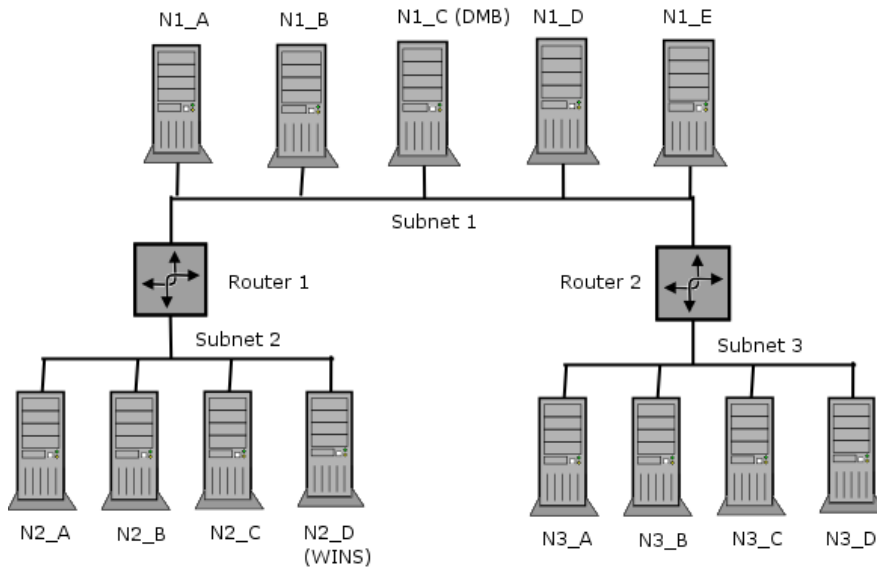


Figure 9.1. Cross-Subnet Browsing Example.

This consists of 3 subnets (1, 2, 3) connected by two routers (R1, R2) which do not pass broadcasts. Subnet 1 has five machines on it, subnet 2 has four machines, subnet 3 has four machines. Assume for the moment that all machines are configured to be in the same workgroup (for simplicity's sake). Machine N1_C on subnet 1 is configured as Domain Master Browser (i.e., it will collate the browse lists for the workgroup). Machine N2_D is configured as WINS server and all the other machines are configured to register their NetBIOS names with it.

As these machines are booted up, elections for master browsers will take place on each of the three subnets. Assume that machine N1_C wins on subnet 1, N2_B wins on subnet 2, and N3_D wins on subnet 3. These machines are known as Local Master Browsers for their particular subnet. N1_C has an advantage in winning as the Local Master Browser on subnet 1 as it is set up as Domain Master Browser.

On each of the three networks, machines that are configured to offer sharing services will broadcast that they are offering these services. The Local Master Browser on each subnet will receive these broadcasts and keep a record of the fact that the machine is offering a service. This list of records is the basis of the browse list. For this case, assume that all the machines are configured to offer services, so all machines will be on the browse list.

For each network, the Local Master Browser on that network is considered “*authoritative*” for all the names it receives via local broadcast. This is because a machine seen by the Local Master Browser via a local broadcast must be on the same network as the Local Master Browser and thus is a “*trusted*” and “*verifiable*” resource. Machines on other networks that the Local Master Browsers learn about when collating their browse lists have not been directly seen. These records are called “*non-authoritative*.”

At this point the browse lists appear as shown in Table 9.1 (these are the machines you

would see in your network neighborhood if you looked in it on a particular network right now).

Table 9.1. Browse Subnet Example 1

Subnet	Browse Master	List
Subnet1	N1_C	N1_A, N1_B, N1_C, N1_D, N1_E
Subnet2	N2_B	N2_A, N2_B, N2_C, N2_D
Subnet3	N3_D	N3_A, N3_B, N3_C, N3_D

At this point all the subnets are separate, and no machine is seen across any of the subnets.

Now examine subnet 2. As soon as N2_B has become the Local Master Browser it looks for a Domain Master Browser with which to synchronize its browse list. It does this by querying the WINS server (N2_D) for the IP address associated with the NetBIOS name WORKGROUP<1B>. This name was registered by the Domain Master Browser (N1_C) with the WINS server as soon as it was started.

Once N2_B knows the address of the Domain Master Browser, it tells it that is the Local Master Browser for subnet 2 by sending a *MasterAnnouncement* packet as a UDP port 138 packet. It then synchronizes with it by doing a *NetServerEnum2* call. This tells the Domain Master Browser to send it all the server names it knows about. Once the Domain Master Browser receives the *MasterAnnouncement* packet, it schedules a synchronization request to the sender of that packet. After both synchronizations are complete the browse lists look as shown in Table 9.2:

Table 9.2. Browse Subnet Example 2

Subnet	Browse Master	List
Subnet1	N1_C	N1_A, N1_B, N1_C, N1_D, N1_E, N2_A(*), N2_B(*), N2_C(*), N2_D(*)
Subnet2	N2_B	N2_A, N2_B, N2_C, N2_D, N1_A(*), N1_B(*), N1_C(*), N1_D(*), N1_E(*)
Subnet3	N3_D	N3_A, N3_B, N3_C, N3_D

Servers with an (*) after them are non-authoritative names.

At this point users looking in their network neighborhood on subnets 1 or 2 will see all the servers on both, users on subnet 3 will still only see the servers on their own subnet.

The same sequence of events that occurred for N2_B now occurs for the Local Master Browser on subnet 3 (N3_D). When it synchronizes browse lists with the Domain Master Browser (N1_A) it gets both the server entries on subnet 1, and those on subnet 2. After N3_D has synchronized with N1_C and vica versa, the browse lists will appear as shown in Table 9.3.

Servers with an (*) after them are non-authoritative names.

At this point, users looking in their network neighborhood on subnets 1 or 3 will see all the servers on all subnets, while users on subnet 2 will still only see the servers on subnets 1 and 2, but not 3.

Finally, the Local Master Browser for subnet 2 (N2_B) will sync again with the Domain Master Browser (N1_C) and will receive the missing server entries. Finally, as when a

Table 9.3. Browse Subnet Example 3

Subnet	Browse Master	List
Subnet1	N1_C	N1_A, N1_B, N1_C, N1_D, N1_E, N2_A(*), N2_B(*), N2_C(*), N2_D(*), N3_A(*), N3_B(*), N3_C(*), N3_D(*)
Subnet2	N2_B	N2_A, N2_B, N2_C, N2_D, N1_A(*), N1_B(*), N1_C(*), N1_D(*), N1_E(*)
Subnet3	N3_D	N3_A, N3_B, N3_C, N3_D, N1_A(*), N1_B(*), N1_C(*), N1_D(*), N1_E(*), N2_A(*), N2_B(*), N2_C(*), N2_D(*)

steady state (if no machines are removed or shut off) has been achieved, the browse lists will appear as shown in Table 9.4.

Table 9.4. Browse Subnet Example 4

Subnet	Browse Master	List
Subnet1	N1_C	N1_A, N1_B, N1_C, N1_D, N1_E, N2_A(*), N2_B(*), N2_C(*), N2_D(*), N3_A(*), N3_B(*), N3_C(*), N3_D(*)
Subnet2	N2_B	N2_A, N2_B, N2_C, N2_D, N1_A(*), N1_B(*), N1_C(*), N1_D(*), N1_E(*), N3_A(*), N3_B(*), N3_C(*), N3_D(*)
Subnet3	N3_D	N3_A, N3_B, N3_C, N3_D, N1_A(*), N1_B(*), N1_C(*), N1_D(*), N1_E(*), N2_A(*), N2_B(*), N2_C(*), N2_D(*)

Servers with an (*) after them are non-authoritative names.

Synchronizations between the Domain Master Browser and Local Master Browsers will continue to occur, but this should remain a steady state operation.

If either router R1 or R2 fails, the following will occur:

1. Names of computers on each side of the inaccessible network fragments will be maintained for as long as 36 minutes in the network neighborhood lists.
2. Attempts to connect to these inaccessible computers will fail, but the names will not be removed from the network neighborhood lists.
3. If one of the fragments is cut off from the WINS server, it will only be able to access servers on its local subnet using subnet-isolated broadcast NetBIOS name resolution. The effects are similar to that of losing access to a DNS server.

9.8 Common Errors

Many questions are asked on the mailing lists regarding browsing. The majority of browsing problems originate from incorrect configuration of NetBIOS name resolution. Some are of particular note.

9.8.1 How Can One Flush the Samba NetBIOS Name Cache without Restarting Samba?

Samba's **nmbd** process controls all browse list handling. Under normal circumstances it is safe to restart **nmbd**. This will effectively flush the Samba NetBIOS name cache and cause it to be rebuilt. This does not make certain that a rogue machine name will not re-appear in the browse list. When **nmbd** is taken out of service, another machine on the network will become the Browse Master. This new list may still have the rogue entry in it. If you really want to clear a rogue machine from the list, every machine on the network will need to be shut down and restarted after all machines are down. Failing a complete restart, the only other thing you can do is wait until the entry times out and is then flushed from the list. This may take a long time on some networks (perhaps months).

9.8.2 Server Resources Can Not Be Listed

“My Client Reports ‘This server is not configured to list shared resources’”

Your guest account is probably invalid for some reason. Samba uses the guest account for browsing in **smbd**. Check that your guest account is valid.

Also see *guest account* in the **smb.conf** man page.

9.8.3 I get an ‘Unable to browse the network’ error

This error can have multiple causes:

- There is no Local Master Browser. Configure **nmbd** or any other machine to serve as Local Master Browser.
- You cannot log onto the machine that is the local master browser. Can you logon to it as a guest user?
- There is no IP connectivity to the Local Master Browser. Can you reach it by broadcast?

9.8.4 Browsing of Shares and Directories is Very Slow

“There are only two machines on a test network. One a Samba server, the other a Windows XP machine. Authentication and logons work perfectly, but when I try to explore shares on the Samba server, the Windows XP client becomes unresponsive. Sometimes it does not respond for some minutes. Eventually, Windows Explorer will respond and displays files and directories without problem. display file and directory.”

*“But, the share is immediately available from a command shell (**cmd**), followed by exploration with **dos** command. Is this a Samba problem or is it a Windows problem? How can I solve this?”*

Here are a few possibilities:

Bad Networking Hardware — Most common defective hardware problems center around low cost or defective HUBs, routers, Network Interface Controllers (NICs) and bad wiring. If one piece of hardware is defective the whole network may suffer. Bad networking hardware can cause data corruption. Most bad networking hardware problems are accompanied by an increase in apparent network traffic, but not all.

The Windows XP WebClient — A number of sites have reported similar slow network browsing problems and found that when the WebClient service is turned off, the problem disappears. This is certainly something that should be explored as it is a simple solution — if it works.

Inconsistent WINS Configuration — This type of problem is common when one client is configured to use a WINS server (that is a TCP/IP configuration setting) and there is no WINS server on the network. Alternately, this will happen if there is a WINS server and Samba is not configured to use it. The use of WINS is highly recommended if the network is using NetBIOS over TCP/IP protocols. If use of NetBIOS over TCP/IP is disabled on all clients, Samba should not be configured as a WINS server neither should it be configured to use one.

Incorrect DNS Configuration — If use of NetBIOS over TCP/IP is disabled, Active Directory is in use and the DNS server has been incorrectly configured. Refer Section 9.3.3 for more information.

ACCOUNT INFORMATION DATABASES

Samba-3 implements a new capability to work concurrently with multiple account backends. The possible new combinations of password backends allows Samba-3 a degree of flexibility and scalability that previously could be achieved only with MS Windows Active Directory. This chapter describes the new functionality and how to get the most out of it.

10.1 Features and Benefits

Samba-3 provides for complete backward compatibility with Samba-2.2.x functionality as follows:

10.1.1 Backward Compatibility Backends

Plain Text — This option uses nothing but the UNIX/Linux `/etc/passwd` style backend. On systems that have Pluggable Authentication Modules (PAM) support, all PAM modules are supported. The behavior is just as it was with Samba-2.2.x, and the protocol limitations imposed by MS Windows clients apply likewise. Please refer to Section 10.2 for more information regarding the limitations of Plain Text password usage.

smbpasswd — This option allows continued use of the `smbpasswd` file that maintains a plain ASCII (text) layout that includes the MS Windows LanMan and NT encrypted passwords as well as a field that stores some account information. This form of password backend does not store any of the MS Windows NT/200x SAM (Security Account Manager) information required to provide the extended controls that are needed for more comprehensive interoperation with MS Windows NT4/200x servers.

This backend should be used only for backward compatibility with older versions of Samba. It may be deprecated in future releases.

ldapsam_compat (Samba-2.2 LDAP Compatibility) — There is a password backend option that allows continued operation with an existing OpenLDAP backend that uses the Samba-2.2.x LDAP schema extension. This option is provided primarily as a migration tool, although there is no reason to force migration at this time. This tool will eventually be deprecated.

Samba-3 introduces a number of new password backend capabilities.

10.1.2 New Backends

tdbsam — This backend provides a rich database backend for local servers. This backend is not suitable for multiple Domain Controllers (i.e., PDC + one or more BDC) installations.

The *tdbsam* password backend stores the old *smbpasswd* information plus the extended MS Windows NT / 200x SAM information into a binary format TDB (trivial database) file. The inclusion of the extended information makes it possible for Samba-3 to implement the same account and system access controls that are possible with MS Windows NT4/200x-based systems.

The inclusion of the *tdbsam* capability is a direct response to user requests to allow simple site operation without the overhead of the complexities of running OpenLDAP. It is recommended to use this only for sites that have fewer than 250 users. For larger sites or implementations, the use of OpenLDAP or of Active Directory integration is strongly recommended.

ldapsam — This provides a rich directory backend for distributed account installation.

Samba-3 has a new and extended LDAP implementation that requires configuration of OpenLDAP with a new format Samba schema. The new format schema file is included in the `examples/LDAP` directory of the Samba distribution.

The new LDAP implementation significantly expands the control abilities that were possible with prior versions of Samba. It is now possible to specify “*per user*” profile settings, home directories, account access controls, and much more. Corporate sites will see that the Samba Team has listened to their requests both for capability and to allow greater scalability.

mysqksam (MySQL based backend) — It is expected that the MySQL-based SAM will be very popular in some corners. This database backend will be of considerable interest to sites that want to leverage existing MySQL technology.

xmlsam (XML based datafile) — Allows the account and password data to be stored in an XML format data file. This backend cannot be used for normal operation, it can only be used in conjunction with **pdbedit**'s `pdb2pdb` functionality. The DTD that is used might be subject to changes in the future.

The *xmbsam* option can be useful for account migration between database backends or backups. Use of this tool will allow the data to be edited before migration into another backend format.

10.2 Technical Information

Old Windows clients send plain text passwords over the wire. Samba can check these passwords by encrypting them and comparing them to the hash stored in the UNIX user database.

Newer Windows clients send encrypted passwords (so-called Lanman and NT hashes) over the wire, instead of plain text passwords. The newest clients will send only encrypted passwords and refuse to send plain text passwords, unless their registry is tweaked.

These passwords can't be converted to UNIX-style encrypted passwords. Because of that, you can't use the standard UNIX user database, and you have to store the Lanman and NT hashes somewhere else.

In addition to differently encrypted passwords, Windows also stores certain data for each user that is not stored in a UNIX user database. For example, workstations the user may logon from, the location where the user's profile is stored, and so on. Samba retrieves and stores this information using a *passdb backend*. Commonly available backends are LDAP, plain text file, and MySQL. For more information, see the man page for `smb.conf` regarding the *passdb backend* parameter.

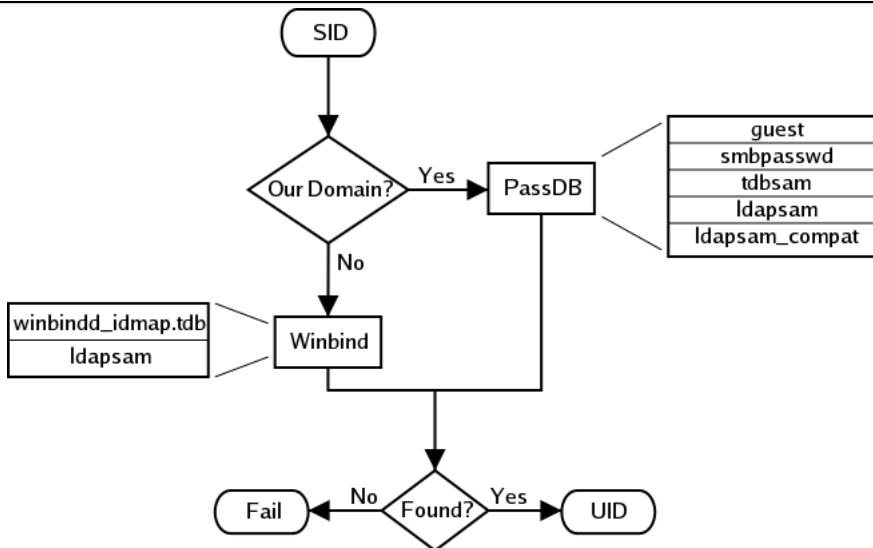


Figure 10.1. IDMAP: Resolution of SIDs to UIDs.

The resolution of SIDs to UIDs is fundamental to correct operation of Samba. In both cases shown, if `winbindd` is not running, or cannot be contacted, then only local SID/UID resolution is possible. See Figure 10.1 and Figure 10.2.

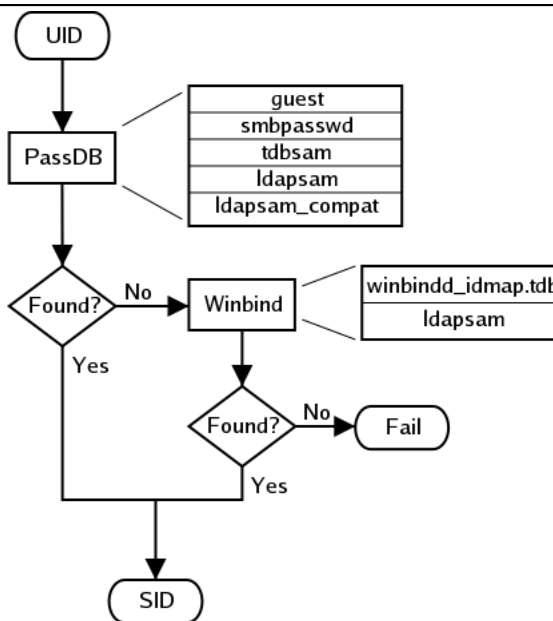


Figure 10.2. IDMAP: Resolution of UIDs to SIDs.

10.2.1 Important Notes About Security

The UNIX and SMB password encryption techniques seem similar on the surface. This similarity is, however, only skin deep. The UNIX scheme typically sends cleartext passwords over the network when logging in. This is bad. The SMB encryption scheme never sends the cleartext password over the network but it does store the 16 byte hashed values on disk. This is also bad. Why? Because the 16 byte hashed values are a “*password equivalent.*” You cannot derive the user’s password from them, but they could potentially be used in a modified client to gain access to a server. This would require considerable technical knowledge on behalf of the attacker but is perfectly possible. You should thus treat the data stored in whatever passdb backend you use (smbpasswd file, LDAP, MYSQL) as though it contained the cleartext passwords of all your users. Its contents must be kept secret and the file should be protected accordingly.

Ideally, we would like a password scheme that involves neither plain text passwords on the network nor on disk. Unfortunately, this is not available as Samba is stuck with having to be compatible with other SMB systems (Windows NT, Windows for Workgroups, Windows 9x/Me).

Windows NT 4.0 Service Pack 3 changed the default setting so plaintext passwords are disabled from being sent over the wire. This mandates either the use of encrypted password support or editing the Windows NT registry to re-enable plaintext passwords.

The following versions of Microsoft Windows do not support full domain security protocols, although they may log onto a domain environment:

- MS DOS Network client 3.0 with the basic network redirector installed.

- Windows 95 with the network redirector update installed.
- Windows 98 [Second Edition].
- Windows Me.

NOTE

MS Windows XP Home does not have facilities to become a Domain Member and it cannot participate in domain logons.

The following versions of MS Windows fully support domain security protocols.

- Windows NT 3.5x.
- Windows NT 4.0.
- Windows 2000 Professional.
- Windows 200x Server/Advanced Server.
- Windows XP Professional.

All current releases of Microsoft SMB/CIFS clients support authentication via the SMB Challenge/Response mechanism described here. Enabling cleartext authentication does not disable the ability of the client to participate in encrypted authentication. Instead, it allows the client to negotiate either plain text or encrypted password handling.

MS Windows clients will cache the encrypted password alone. Where plain text passwords are re-enabled through the appropriate registry change, the plain text password is never cached. This means that in the event that a network connections should become disconnected (broken), only the cached (encrypted) password will be sent to the resource server to effect an auto-reconnect. If the resource server does not support encrypted passwords the auto-reconnect will fail. Use of encrypted passwords is strongly advised.

10.2.1.1 Advantages of Encrypted Passwords

- Plaintext passwords are not passed across the network. Someone using a network sniffer cannot just record passwords going to the SMB server.
- Plaintext passwords are not stored anywhere in memory or on disk.
- Windows NT does not like talking to a server that does not support encrypted passwords. It will refuse to browse the server if the server is also in User Level security mode. It will insist on prompting the user for the password on each connection, which is very annoying. The only things you can do to stop this is to use SMB encryption.
- Encrypted password support allows automatic share (resource) reconnects.
- Encrypted passwords are essential for PDC/BDC operation.

10.2.1.2 Advantages of Non-Encrypted Passwords

- Plaintext passwords are not kept on disk, and are not cached in memory.
- Uses same password file as other UNIX services such as Login and FTP.
- Use of other services (such as Telnet and FTP) that send plain text passwords over the network, so sending them for SMB is not such a big deal.

10.2.2 Mapping User Identifiers between MS Windows and UNIX

Every operation in UNIX/Linux requires a user identifier (UID), just as in MS Windows NT4/200x this requires a Security Identifier (SID). Samba provides two means for mapping an MS Windows user to a UNIX/Linux UID.

First, all Samba SAM (Security Account Manager database) accounts require a UNIX/Linux UID that the account will map to. As users are added to the account information database, Samba will call the *add user script* interface to add the account to the Samba host OS. In essence all accounts in the local SAM require a local user account.

The second way to effect Windows SID to UNIX UID mapping is via the *idmap uid* and *idmap gid* parameters in *smb.conf*. Please refer to the man page for information about these parameters. These parameters are essential when mapping users from a remote SAM server.

10.2.3 Mapping Common UIDs/GIDs on Distributed Machines

Samba-3 has a special facility that makes it possible to maintain identical UIDs and GIDs on all servers in a distributed network. A distributed network is one where there exists a PDC, one or more BDCs and/or one or more Domain Member servers. Why is this important? This is important if files are being shared over more than one protocol (e.g., NFS) and where users are copying files across UNIX/Linux systems using tools such as **rsync**.

The special facility is enabled using a parameter called *idmap backend*. The default setting for this parameter is an empty string. Technically it is possible to use an LDAP based idmap backend for UIDs and GIDs, but it makes most sense when this is done for network configurations that also use LDAP for the SAM backend. A sample use is shown in Example 10.1.

Example 10.1. Example configuration with the LDAP idmap backend

```
[global]
idmap backend = ldap:ldap://ldap-server.kenya.org:636
idmap backend = ldap:ldaps://ldap-server.kenya.org
```

A network administrator who wants to make significant use of LDAP backends will sooner or later be exposed to the excellent work done by PADL Software. PADL <http://www.padl.com>¹ have produced and released to open source an array of tools that might be of

¹<http://www.padl.com>

interest. These tools include:

- *nss_ldap*: An LDAP Name Service Switch module to provide native name service support for AIX, Linux, Solaris, and other operating systems. This tool can be used for centralized storage and retrieval of UIDs/GIDs.
- *pam_ldap*: A PAM module that provides LDAP integration for UNIX/Linux system access authentication.
- *idmap_ad*: An IDMAP backend that supports the Microsoft Services for UNIX RFC 2307 schema available from their web site².

10.3 Account Management Tools

Samba provides two tools for management of user and machine accounts. These tools are called **smbpasswd** and **pdbedit**. A third tool is under development but is not expected to ship in time for Samba-3.0.0. The new tool will be a TCL/TK GUI tool that looks much like the MS Windows NT4 Domain User Manager. Hopefully this will be announced in time for the Samba-3.0.1 release.

10.3.1 The *smbpasswd* Command

The **smbpasswd** utility is similar to the **passwd** or **yppasswd** programs. It maintains the two 32 byte password fields in the **passdb** backend.

smbpasswd works in a client-server mode where it contacts the local **smbd** to change the user's password on its behalf. This has enormous benefits.

smbpasswd has the capability to change passwords on Windows NT servers (this only works when the request is sent to the NT Primary Domain Controller if changing an NT Domain user's password).

smbpasswd can be used to:

- *add* user or machine accounts.
- *delete* user or machine accounts.
- *enable* user or machine accounts.
- *disable* user or machine accounts.
- *set to NULL* user passwords.
- *manage interdomain trust accounts*.

To run **smbpasswd** as a normal user just type:

```
$ smbpasswd
Old SMB password: secret
```

²http://www.padl.com/download/xad_loss_plugins.tar.gz

For *secret*, type old value here or press return if there is no old password.

```
New SMB Password: new secret
Repeat New SMB Password: new secret
```

If the old value does not match the current value stored for that user, or the two new values do not match each other, then the password will not be changed.

When invoked by an ordinary user, the command will only allow the user to change his or her own SMB password.

When run by root, **smbpasswd** may take an optional argument specifying the user name whose SMB password you wish to change. When run as root, **smbpasswd** does not prompt for or check the old password value, thus allowing root to set passwords for users who have forgotten their passwords.

smbpasswd is designed to work in the way familiar to UNIX users who use the **passwd** or **yppasswd** commands. While designed for administrative use, this tool provides essential User Level password change capabilities.

For more details on using **smbpasswd**, refer to the man page (the definitive reference).

10.3.2 The *pdbedit* Command

pdbedit is a tool that can be used only by root. It is used to manage the passdb backend. **pdbedit** can be used to:

- add, remove or modify user accounts.
- list user accounts.
- migrate user accounts.

The **pdbedit** tool is the only one that can manage the account security and policy settings. It is capable of all operations that **smbpasswd** can do as well as a super set of them.

One particularly important purpose of the **pdbedit** is to allow the migration of account information from one passdb backend to another. See the XML password backend section of this chapter.

The following is an example of the user account information that is stored in a tdbsam password backend. This listing was produced by running:

```
$ pdbedit -Lv met
UNIX username:      met
NT username:
Account Flags:     [UX          ]
User SID:          S-1-5-21-1449123459-1407424037-3116680435-2004
Primary Group SID: S-1-5-21-1449123459-1407424037-3116680435-1201
Full Name:         Melissa E Terpstra
Home Directory:    \\frodo\met\Win9Profile
```



```

HomeDir Drive:      H:
Logon Script:       scripts\logon.bat
Profile Path:       \\frodo\Profiles\met
Domain:             MIDEARTH
Account desc:
Workstations:       melbelle
Munged dial:
Logon time:         0
Logoff time:        Mon, 18 Jan 2038 20:14:07 GMT
Kickoff time:       Mon, 18 Jan 2038 20:14:07 GMT
Password last set:  Sat, 14 Dec 2002 14:37:03 GMT
Password can change: Sat, 14 Dec 2002 14:37:03 GMT
Password must change: Mon, 18 Jan 2038 20:14:07 GMT

```

The **pdbedit** tool allows migration of authentication (account) databases from one backend to another. For example: To migrate accounts from an old **smbpasswd** database to a **tdbsam** backend:

1. Set the *passdb backend* = *tdbsam*, *smbpasswd*.
2. Execute:

```
root# pdbedit -i smbpasswd -e tdbsam
```

3. Now remove the *smbpasswd* from the *passdb* backend configuration in *smb.conf*.

10.4 Password Backends

Samba offers the greatest flexibility in backend account database design of any SMB/CIFS server technology available today. The flexibility is immediately obvious as one begins to explore this capability.

It is possible to specify not only multiple different password backends, but even multiple backends of the same type. For example, to use two different *tdbsam* databases:

```

passdb backend = tdbsam:/etc/samba/passdb.tdb \
tdbsam:/etc/samba/old-passdb.tdb

```

10.4.1 Plaintext

Older versions of Samba retrieved user information from the UNIX user database and eventually some other fields from the file */etc/samba/smbpasswd* or */etc/smbpasswd*. When password encryption is disabled, no SMB specific data is stored at all. Instead all operations are conducted via the way that the Samba host OS will access its */etc/passwd* database. Linux systems For example, all operations are done via PAM.

10.4.2 smbpasswd — Encrypted Password Database

Traditionally, when configuring *encrypt passwords* = yes in Samba's `smb.conf` file, user account information such as username, LM/NT password hashes, password change times, and account flags have been stored in the `smbpasswd(5)` file. There are several disadvantages to this approach for sites with large numbers of users (counted in the thousands).

- The first problem is that all lookups must be performed sequentially. Given that there are approximately two lookups per domain logon (one for a normal session connection such as when mapping a network drive or printer), this is a performance bottleneck for large sites. What is needed is an indexed approach such as used in databases.
- The second problem is that administrators who desire to replicate a `smbpasswd` file to more than one Samba server were left to use external tools such as `rsync(1)` and `ssh(1)` and wrote custom, in-house scripts.
- Finally, the amount of information that is stored in an `smbpasswd` entry leaves no room for additional attributes such as a home directory, password expiration time, or even a Relative Identifier (RID).

As a result of these deficiencies, a more robust means of storing user attributes used by `smbd` was developed. The API which defines access to user accounts is commonly referred to as the `samdb` interface (previously this was called the `passdb` API, and is still so named in the Samba CVS trees).

Samba provides an enhanced set of `passdb` backends that overcome the deficiencies of the `smbpasswd` plain text database. These are `tdbsam`, `ldapsam` and `xmlsam`. Of these, `ldapsam` will be of most interest to large corporate or enterprise sites.

10.4.3 tdbsam

Samba can store user and machine account data in a “*TDB*” (Trivial Database). Using this backend does not require any additional configuration. This backend is recommended for new installations that do not require LDAP.

As a general guide, the Samba Team does not recommend using the `tdbsam` backend for sites that have 250 or more users. Additionally, `tdbsam` is not capable of scaling for use in sites that require PDB/BDC implementations that require replication of the account database. Clearly, for reason of scalability, the use of `ldapsam` should be encouraged.

The recommendation of a 250 user limit is purely based on the notion that this would generally involve a site that has routed networks, possibly spread across more than one physical location. The Samba Team has not at this time established the performance based scalability limits of the `tdbsam` architecture.

10.4.4 ldapsam

There are a few points to stress that the `ldapsam` does not provide. The LDAP support referred to in this documentation does not include:

- A means of retrieving user account information from an Windows 200x Active Directory server.
- A means of replacing `/etc/passwd`.

The second item can be accomplished by using LDAP NSS and PAM modules. LGPL versions of these libraries can be obtained from PADL Software³. More information about the configuration of these packages may be found at *LDAP, System Administration*; Gerald Carter by O'Reilly; Chapter 6: Replacing NIS.⁴

This document describes how to use an LDAP directory for storing Samba user account information traditionally stored in the `smbpasswd(5)` file. It is assumed that the reader already has a basic understanding of LDAP concepts and has a working directory server already installed. For more information on LDAP architectures and directories, please refer to the following sites:

- OpenLDAP⁵
- Sun iPlanet Directory Server⁶

Two additional Samba resources which may prove to be helpful are:

- The Samba-PDC-LDAP-HOWTO⁷ maintained by Ignacio Coupeau.
- The NT migration scripts from IDEALX⁸ that are geared to manage users and group in such a Samba-LDAP Domain Controller configuration.

10.4.4.1 Supported LDAP Servers

The LDAP `ldapsam` code has been developed and tested using the OpenLDAP 2.0 and 2.1 server and client libraries. The same code should work with Netscape's Directory Server and client SDK. However, there are bound to be compile errors and bugs. These should not be hard to fix. Please submit fixes via the process outlined in Chapter 34, *Reporting Bugs*.

10.4.4.2 Schema and Relationship to the RFC 2307 `posixAccount`

Samba-3.0 includes the necessary schema file for OpenLDAP 2.0 in `examples/LDAP/samba.schema`. The `sambaSamAccount` objectclass is given here:

```
objectclass (1.3.6.1.4.1.7165.2.2.6 NAME 'sambaSamAccount' SUP top AUXILIARY
DESC 'Samba-3.0 Auxiliary SAM Account'
MUST ( uid $ sambaSID )
MAY ( cn $ sambaLMPassword $ sambaNTPassword $ sambaPwdLastSet $
      sambaLogonTime $ sambaLogoffTime $ sambaKickoffTime $
      sambaPwdCanChange $ sambaPwdMustChange $ sambaAcctFlags $
```

³<http://www.padl.com/>

⁴<http://safari.oreilly.com/?XmlId=1-56592-491-6>

⁵<http://www.openldap.org/>

⁶<http://iplanet.netscape.com/directory>

⁷<http://www.unav.es/cti/ldap-smb/ldap-smb-3-howto.html>

⁸<http://samba.idealx.org/>

```

displayName $ sambaHomePath $ sambaHomeDrive $ sambaLogonScript $
sambaProfilePath $ description $ sambaUserWorkstations $
sambaPrimaryGroupSID $ sambaDomainName ))

```

The `samba.schema` file has been formatted for OpenLDAP 2.0/2.1. The Samba Team owns the OID space used by the above schema and recommends its use. If you translate the schema to be used with Netscape DS, please submit the modified schema file as a patch to jerry@samba.org⁹.

Just as the `smbpasswd` file is meant to store information that provides information additional to a user's `/etc/passwd` entry, so is the `sambaSamAccount` object meant to supplement the UNIX user account information. A `sambaSamAccount` is a `AUXILIARY` objectclass so it can be used to augment existing user account information in the LDAP directory, thus providing information needed for Samba account handling. However, there are several fields (e.g., `uid`) that overlap with the `posixAccount` objectclass outlined in RFC2307. This is by design.

In order to store all user account information (UNIX and Samba) in the directory, it is necessary to use the `sambaSamAccount` and `posixAccount` objectclasses in combination. However, `smbd` will still obtain the user's UNIX account information via the standard C library calls (e.g., `getpwnam()`, et al). This means that the Samba server must also have the LDAP NSS library installed and functioning correctly. This division of information makes it possible to store all Samba account information in LDAP, but still maintain UNIX account information in NIS while the network is transitioning to a full LDAP infrastructure.

10.4.4.3 OpenLDAP Configuration

To include support for the `sambaSamAccount` object in an OpenLDAP directory server, first copy the `samba.schema` file to `slapd`'s configuration directory. The `samba.schema` file can be found in the directory `examples/LDAP` in the Samba source distribution.

```
root# cp samba.schema /etc/openldap/schema/
```

Next, include the `samba.schema` file in `slapd.conf`. The `sambaSamAccount` object contains two attributes that depend on other schema files. The `uid` attribute is defined in `cosine.schema` and the `displayName` attribute is defined in the `inetorgperson.schema` file. Both of these must be included before the `samba.schema` file.

```

## /etc/openldap/slapd.conf

## schema files (core.schema is required by default)
include          /etc/openldap/schema/core.schema

## needed for sambaSamAccount
include          /etc/openldap/schema/cosine.schema
include          /etc/openldap/schema/inetorgperson.schema

```

⁹<mailto:jerry@samba.org>

```
include          /etc/openldap/schema/samba.schema
include          /etc/openldap/schema/nis.schema
....
```

It is recommended that you maintain some indices on some of the most useful attributes, as in the following example, to speed up searches made on sambaSamAccount objectclasses (and possibly posixAccount and posixGroup as well):

```
# Indices to maintain
## required by OpenLDAP
index objectclass          eq

index cn                   pres,sub,eq
index sn                   pres,sub,eq
## required to support pdb_getsampwnam
index uid                  pres,sub,eq
## required to support pdb_getsambapwrid()
index displayName         pres,sub,eq

## uncomment these if you are storing posixAccount and
## posixGroup entries in the directory as well
##index uidNumber         eq
##index gidNumber         eq
##index memberUid         eq

index sambaSID             eq
index sambaPrimaryGroupSID eq
index sambaDomainName     eq
index default              sub
```

Create the new index by executing:

```
root# ./sbin/slapindex -f slapd.conf
```

Remember to restart slapd after making these changes:

```
root# /etc/init.d/slapd restart
```

10.4.4.4 Initialize the LDAP Database

Before you can add accounts to the LDAP database you must create the account containers that they will be stored in. The following LDIF file should be modified to match your needs (DNS entries, and so on):

```
# Organization for Samba Base
dn: dc=kenya,dc=org
objectclass: dcObject
objectclass: organization
dc: kenya
o: Kenya Org Network
description: The Samba-3 Network LDAP Example

# Organizational Role for Directory Management
dn: cn=Manager,dc=kenya,dc=org
objectclass: organizationalRole
cn: Manager
description: Directory Manager

# Setting up container for users
dn: ou=People,dc=kenya,dc=org
objectclass: top
objectclass: organizationalUnit
ou: People

# Setting up admin handle for People OU
dn: cn=admin,ou=People,dc=kenya,dc=org
cn: admin
objectclass: top
objectclass: organizationalRole
objectclass: simpleSecurityObject
userPassword: {SSHA}c3ZM9tBaBo9autm1dL3waDS21+JSfQVz

# Setting up container for groups
dn: ou=Groups,dc=kenya,dc=org
objectclass: top
objectclass: organizationalUnit
ou: Groups

# Setting up admin handle for Groups OU
dn: cn=admin,ou=Groups,dc=kenya,dc=org
cn: admin
objectclass: top
objectclass: organizationalRole
objectclass: simpleSecurityObject
userPassword: {SSHA}c3ZM9tBaBo9autm1dL3waDS21+JSfQVz

# Setting up container for computers
dn: ou=Computers,dc=kenya,dc=org
objectclass: top
objectclass: organizationalUnit
ou: Computers
```

```
# Setting up admin handle for Computers OU
dn: cn=admin,ou=Computers,dc=kenya,dc=org
cn: admin
objectclass: top
objectclass: organizationalRole
objectclass: simpleSecurityObject
userPassword: {SSHA}c3ZM9tBaBo9autm1dL3waDS21+JSfQVz
```

The userPassword shown above should be generated using **slappasswd**.

The following command will then load the contents of the LDIF file into the LDAP database.

```
$ slapadd -v -l initldap.dif
```

Do not forget to secure your LDAP server with an adequate access control list as well as an admin password.

NOTE



Before Samba can access the LDAP server you need to store the LDAP admin password into the Samba-3 `secrets.tdb` database by:

```
root# smbpasswd -w secret
```

10.4.4.5 Configuring Samba

The following parameters are available in `smb.conf` only if your version of Samba was built with LDAP support. Samba automatically builds with LDAP support if the LDAP libraries are found.

LDAP related `smb.conf` options: *passdb backend = ldapsam:url, ldap admin dn, ldap delete dn, ldap filter, ldap group suffix, ldap idmap suffix, ldap machine suffix, ldap passwd sync, ldap ssl, ldap suffix, ldap user suffix,*

These are described in the `smb.conf` man page and so will not be repeated here. However, a sample `smb.conf` file for use with an LDAP directory could appear as shown in Example 10.2.

10.4.4.6 Accounts and Groups Management

As user accounts are managed through the `sambaSamAccount` objectclass, you should modify your existing administration tools to deal with `sambaSamAccount` attributes.

Example 10.2. Configuration with LDAP

```

[global]
security = user
encrypt passwords = yes
netbios name = MORIA
workgroup = NOLDOR
# ldap related parameters
# define the DN to use when binding to the directory servers
# The password for this DN is not stored in smb.conf. Rather it
# must be set by using 'smbpasswd -w secretpw' to store the
# passphrase in the secrets.tdb file. If the "ldap admin dn" values
# change, this password will need to be reset.
ldap admin dn = "cn=Manager,ou=People,dc=quenya,dc=org"
# Define the SSL option when connecting to the directory
# ('off', 'start tls', or 'on' (default))
ldap ssl = start tls
# syntax: passdb backend = ldapsam:ldap://server-name[:port]
passdb backend = ldapsam:ldap://frodo.quenya.org
# smbpasswd -x delete the entire dn-entry
ldap delete dn = no
# the machine and user suffix added to the base suffix
# wrote WITHOUT quotes. NULL suffixes by default
ldap user suffix = ou=People
ldap group suffix = ou=Groups
ldap machine suffix = ou=Computers
# Trust UNIX account information in LDAP
# (see the smb.conf manpage for details)
# specify the base DN to use when searching the directory
ldap suffix = dc=quenya,dc=org
# generally the default ldap search filter is ok
ldap filter = (&(uid=%u)(objectclass=sambaSamAccount))

```

Machine accounts are managed with the `sambaSamAccount` objectclass, just like users accounts. However, it is up to you to store those accounts in a different tree of your LDAP namespace. You should use “`ou=Groups,dc=quenya,dc=org`” to store groups and “`ou=People,dc=quenya,dc=org`” to store users. Just configure your NSS and PAM accordingly (usually, in the `/etc/openldap/sldap.conf` configuration file).

In Samba-3, the group management system is based on POSIX groups. This means that Samba makes use of the `posixGroup` objectclass. For now, there is no NT-like group system management (global and local groups). Samba-3 knows only about **Domain Groups** and, unlike MS Windows 2000 and Active Directory, Samba-3 does not support nested groups.

10.4.4.7 Security and sambaSamAccount

There are two important points to remember when discussing the security of sambaSamAccount entries in the directory.

- *Never* retrieve the lmPassword or ntPassword attribute values over an unencrypted LDAP session.
- *Never* allow non-admin users to view the lmPassword or ntPassword attribute values.

These password hashes are cleartext equivalents and can be used to impersonate the user without deriving the original cleartext strings. For more information on the details of LM/NT password hashes, refer to the Account Information Database section of this chapter.

To remedy the first security issue, the `ldap ssl smb.conf` parameter defaults to require an encrypted session (`ldap ssl = on`) using the default port of 636 when contacting the directory server. When using an OpenLDAP server, it is possible to use the use the Start-TLS LDAP extended operation in the place of LDAPS. In either case, you are strongly discouraged to disable this security (`ldap ssl = off`).

Note that the LDAPS protocol is deprecated in favor of the LDAPv3 StartTLS extended operation. However, the OpenLDAP library still provides support for the older method of securing communication between clients and servers.

The second security precaution is to prevent non-administrative users from harvesting password hashes from the directory. This can be done using the following ACL in `slapd.conf`:

```
## allow the "ldap admin dn" access, but deny everyone else
access to attrs=lmPassword,ntPassword
    by dn="cn=Samba Admin,ou=People,dc=quenya,dc=org" write
    by * none
```

10.4.4.8 LDAP Special Attributes for sambaSamAccounts

The sambaSamAccount objectclass is composed of the attributes shown in Table 10.1, and Table 10.2.

The majority of these parameters are only used when Samba is acting as a PDC of a domain (refer to Chapter 4, *Domain Control*, for details on how to configure Samba as a Primary Domain Controller). The following four attributes are only stored with the sambaSamAccount entry if the values are non-default values:

- sambaHomePath
- sambaLogonScript
- sambaProfilePath
- sambaHomeDrive

These attributes are only stored with the sambaSamAccount entry if the values are non-default values. For example, assume MORIA has now been configured as a PDC and that

logon home = `\\%L%\%u` was defined in its `smb.conf` file. When a user named “*becky*” logs on to the domain, the *logon home* string is expanded to `\\MORIA\becky`. If the `smb-Home` attribute exists in the entry “*uid=becky,ou=People,dc=samba,dc=org*”, this value is used. However, if this attribute does not exist, then the value of the *logon home* parameter is used in its place. Samba will only write the attribute value to the directory entry if the value is something other than the default (e.g., `\\MOBY\becky`).

10.4.4.9 Example LDIF Entries for a `sambaSamAccount`

The following is a working LDIF that demonstrates the use of the `SambaSamAccount` objectclass:

```
dn: uid=guest2, ou=People,dc=quenya,dc=org
sambaLMPassword: 878D8014606CDA29677A44EFA1353FC7
sambaPwdMustChange: 2147483647
sambaPrimaryGroupSID: S-1-5-21-2447931902-1787058256-3961074038-513
sambaNTPassword: 552902031BEDE9EFAAD3B435B51404EE
sambaPwdLastSet: 1010179124
sambaLogonTime: 0
objectClass: sambaSamAccount
uid: guest2
sambaKickoffTime: 2147483647
sambaAcctFlags: [UX          ]
sambaLogoffTime: 2147483647
sambaSID: S-1-5-21-2447931902-1787058256-3961074038-5006
sambaPwdCanChange: 0
```

The following is an LDIF entry for using both the `sambaSamAccount` and `posixAccount` objectclasses:

```
dn: uid=gcarter, ou=People,dc=quenya,dc=org
sambaLogonTime: 0
displayName: Gerald Carter
sambaLMPassword: 552902031BEDE9EFAAD3B435B51404EE
sambaPrimaryGroupSID: S-1-5-21-2447931902-1787058256-3961074038-1201
objectClass: posixAccount
objectClass: sambaSamAccount
sambaAcctFlags: [UX          ]
userPassword: {crypt}BpM2ej8Rkzogo
uid: gcarter
uidNumber: 9000
cn: Gerald Carter
loginShell: /bin/bash
logoffTime: 2147483647
gidNumber: 100
sambaKickoffTime: 2147483647
```

```
sambaPwdLastSet: 1010179230
sambaSID: S-1-5-21-2447931902-1787058256-3961074038-5004
homeDirectory: /home/moria/gcarter
sambaPwdCanChange: 0
sambaPwdMustChange: 2147483647
sambaNTPassword: 878D8014606CDA29677A44EFA1353FC7
```

10.4.4.10 Password Synchronization

Samba-3 and later can update the non-samba (LDAP) password stored with an account. When using `pam_ldap`, this allows changing both UNIX and Windows passwords at once.

The `ldap passwd sync` options can have the values shown in Table 10.3.

More information can be found in the `smb.conf` manpage.

10.4.5 MySQL

Every so often someone will come along with a great new idea. Storing user accounts in a SQL backend is one of them. Those who want to do this are in the best position to know what the specific benefits are to them. This may sound like a cop-out, but in truth we cannot attempt to document every little detail why certain things of marginal utility to the bulk of Samba users might make sense to the rest. In any case, the following instructions should help the determined SQL user to implement a working system.

10.4.5.1 Creating the Database

You can set up your own table and specify the field names to `pdb_mysql` (see below for the column names) or use the default table. The file `examples/pdb/mysql/mysql.dump` contains the correct queries to create the required tables. Use the command:

```
$ mysql -username -hhostname -ppassword \  
    databasename < /path/to/samba/examples/pdb/mysql/mysql.dump
```

10.4.5.2 Configuring

This plugin lacks some good documentation, but here is some brief information. Add the following to the `passdb backend` variable in your `smb.conf`:

```
passdb backend = [other-plugins] mysql:identifier [other-plugins]
```

The identifier can be any string you like, as long as it does not collide with the identifiers of other plugins or other instances of `pdb_mysql`. If you specify multiple `pdb_mysql.so` entries in `passdb backend`, you also need to use different identifiers.

Additional options can be given through the `smb.conf` file in the `[global]` section. Refer to Table 10.4.

WARNING



Since the password for the MySQL user is stored in the `smb.conf` file, you should make the `smb.conf` file readable only to the user who runs Samba. This is considered a security bug and will soon be fixed.

Names of the columns are given in Table 10.5. The default column names can be found in the example table dump.

You can put a colon (:) after the name of each column, which should specify the column to update when updating the table. You can also specify nothing behind the colon. Then the field data will not be updated. Setting a column name to `NULL` means the field should not be used.

An example configuration can be found in Example 10.3.

Example 10.3. Example configuration for the MySQL passdb backend

```
[global]
passdb backend = mysql:foo
foo:mysql user = samba
foo:mysql password = abmas
foo:mysql database = samba
# domain name is static and can't be changed
foo:domain column = 'MYWORKGROUP':
# The fullname column comes from several other columns
foo:fullname column = CONCAT(firstname, ' ', surname):
# Samba should never write to the password columns
foo:lanman pass column = lm_pass:
foo:nt pass column = nt_pass:
# The unknown 3 column is not stored
foo:unknown 3 column = NULL
```

10.4.5.3 Using Plaintext Passwords or Encrypted Password

I strongly discourage the use of plaintext passwords, however, you can use them.

If you would like to use plaintext passwords, set 'identifier:lanman pass column' and 'identifier:nt pass column' to 'NULL' (without the quotes) and 'identifier:plain pass column' to the name of the column containing the plaintext passwords.

If you use encrypted passwords, set the 'identifier:plain pass column' to 'NULL' (without the quotes). This is the default.

10.4.5.4 Getting Non-Column Data from the Table

It is possible to have not all data in the database by making some ‘constant’.

For example, you can set ‘identifier:fullname column’ to something like `CONCAT(Firstname, ’ ’, Surname)`

Or, set ‘identifier:workstations column’ to: `NULL`

See the MySQL documentation for more language constructs.

10.4.6 XML

This module requires libxml2 to be installed.

The usage of `pdb_xml` is fairly straightforward. To export data, use:

```
$ pdbedit -e xml:filename
```

(where filename is the name of the file to put the data in)

To import data, use: `$ pdbedit -i xml:filename`

10.5 Common Errors

10.5.1 Users Cannot Logon

“I’ve installed Samba, but now I can’t log on with my UNIX account!”

Make sure your user has been added to the current Samba *passdb backend*. Read the section Section 10.3 for details.

10.5.2 Users Being Added to the Wrong Backend Database

A few complaints have been received from users that just moved to Samba-3. The following `smb.conf` file entries were causing problems, new accounts were being added to the old `smbpasswd` file, not to the `tdbsam passdb.tdb` file:

```
...  
passdb backend = smbpasswd, tdbsam  
...
```

Samba will add new accounts to the first entry in the *passdb backend* parameter entry. If you want to update to the `tdbsam`, then change the entry to:

```
passdb backend = tdbsam, smbpasswd
```

10.5.3 Configuration of auth methods

When explicitly setting an *auth methods* parameter, *guest* must be specified as the first entry on the line, for example, `auth methods = guest sam`.

This is the exact opposite of the requirement for the *passwd backend* option, where it must be the *LAST* parameter on the line.

Table 10.1. Attributes in the `sambaSamAccount` objectclass (LDAP) — Part A

<code>sambaLMPassword</code>	The LANMAN password 16-byte hash stored as a character representation of a hexadecimal string.
<code>sambaNTPassword</code>	The NT password hash 16-byte stored as a character representation of a hexadecimal string.
<code>sambaPwdLastSet</code>	The integer time in seconds since 1970 when the <code>sambaLMPassword</code> and <code>sambaNTPassword</code> attributes were last set.
<code>sambaAcctFlags</code>	String of 11 characters surrounded by square brackets [] representing account flags such as U (user), W (workstation), X (no password expiration), I (Domain trust account), H (Home dir required), S (Server trust account), and D (disabled).
<code>sambaLogonTime</code>	Integer value currently unused
<code>sambaLogoffTime</code>	Integer value currently unused
<code>sambaKickoffTime</code>	Specifies the time (UNIX time format) when the user will be locked down and cannot login any longer. If this attribute is omitted, then the account will never expire. If you use this attribute together with ‘shadowExpire’ of the ‘shadowAccount’ objectClass, will enable accounts to expire completely on an exact date.
<code>sambaPwdCanChange</code>	Specifies the time (UNIX time format) from which on the user is allowed to change his password. If attribute is not set, the user will be free to change his password whenever he wants.
<code>sambaPwdMustChange</code>	Specifies the time (UNIX time format) since when the user is forced to change his password. If this value is set to ‘0’, the user will have to change his password at first login. If this attribute is not set, then the password will never expire.
<code>sambaHomeDrive</code>	Specifies the drive letter to which to map the UNC path specified by <code>sambaHomePath</code> . The drive letter must be specified in the form “X:” where X is the letter of the drive to map. Refer to the “ <i>logon drive</i> ” parameter in the <code>smb.conf(5)</code> man page for more information.
<code>sambaLogonScript</code>	The <code>sambaLogonScript</code> property specifies the path of the user’s logon script, .CMD, .EXE, or .BAT file. The string can be null. The path is relative to the netlogon share. Refer to the <i>logon script</i> parameter in the <code>smb.conf</code> man page for more information.
<code>sambaProfilePath</code>	Specifies a path to the user’s profile. This value can be a null string, a local absolute path, or a UNC path. Refer to the <i>logon path</i> parameter in the <code>smb.conf</code> man page for more information.
<code>sambaHomePath</code>	The <code>sambaHomePath</code> property specifies the path of the home directory for the user. The string can be null. If <code>sambaHomeDrive</code> is set and specifies a drive letter, <code>sambaHomePath</code> should be a UNC path. The path must be a network UNC path of the form <code>\\server\share\directory</code> . This value can be a null string. Refer to the logon home parameter in the <code>smb.conf</code> man page for more information.

Table 10.2. Attributes in the sambaSamAccount objectclass (LDAP) — Part B

sambaUserWorkstations	Here you can give a comma-separated list of machines on which the user is allowed to login. You may observe problems when you try to connect to an Samba Domain Member. Because Domain Members are not in this list, the Domain Controllers will reject them. Where this attribute is omitted, the default implies no restrictions.
sambaSID	The security identifier(SID) of the user. The Windows equivalent of UNIX UIDs.
sambaPrimaryGroupSID	The Security IDentifier (SID) of the primary group of the user.
sambaDomainName	Domain the user is part of.

Table 10.3. Possible *ldap passwd sync* values

Value	Description
yes	When the user changes his password, update <code>ntPassword</code> , <code>lmPassword</code> and the <code>password</code> fields.
no	Only update <code>ntPassword</code> and <code>lmPassword</code> .
only	Only update the LDAP password and let the LDAP server worry about the other fields. This option is only available on some LDAP servers. Only when the LDAP server supports <code>LDAP_EXOP_X_MODIFY_PASSWD</code> .

Table 10.4. Basic smb.conf options for MySQL passwd backend

Field	Contents
mysql host	Host name, defaults to 'localhost'
mysql password	
mysql user	Defaults to 'samba'
mysql database	Defaults to 'samba'
mysql port	Defaults to 3306
table	Name of the table containing the users

Table 10.5. MySQL field names for MySQL passwd backend

Field	Type	Contents
logon time column	int(9)	UNIX time stamp of last logon of user
logoff time column	int(9)	UNIX time stamp of last logoff of user
kickoff time column	int(9)	UNIX time stamp of moment user should be kicked off workstation (not enforced)
pass last set time column	int(9)	UNIX time stamp of moment password was last set
pass can change time column	int(9)	UNIX time stamp of moment from which password can be changed
pass must change time column	int(9)	UNIX time stamp of moment on which password must be changed
username column	varchar(255)	UNIX username
domain column	varchar(255)	NT domain user belongs to
nt username column	varchar(255)	NT username
fullname column	varchar(255)	Full name of user
home dir column	varchar(255)	UNIX homedir path
dir drive column	varchar(2)	Directory drive path (e.g., "H:")
logon script column	varchar(255)	Batch file to run on client side when logging on
profile path column	varchar(255)	Path of profile
acct desc column	varchar(255)	Some ASCII NT user data
workstations column	varchar(255)	Workstations user can logon to (or NULL for all)
unknown string column	varchar(255)	Unknown string
munged dial column	varchar(255)	Unknown
user sid column	varchar(255)	NT user SID
group sid column	varchar(255)	NT group SID
lanman pass column	varchar(255)	Encrypted lanman password
nt pass column	varchar(255)	Encrypted nt passwd
plain pass column	varchar(255)	Plaintext password
acct ctrl column	int(9)	NT user data
unknown 3 column	int(9)	Unknown
logon divs column	int(9)	Unknown
hours len column	int(9)	Unknown
bad password count column	int(5)	Number of failed password tries before disabling an account
logon count column	int(5)	Number of logon attempts
unknown 6 column	int(9)	Unknown

GROUP MAPPING — MS WINDOWS AND UNIX

Starting with Samba-3, new group mapping functionality is available to create associations between Windows group SIDs and UNIX groups. The **groupmap** subcommand included with the `net` tool can be used to manage these associations.

The new facility for mapping NT Groups to UNIX system groups allows the administrator to decide which NT Domain Groups are to be exposed to MS Windows clients. Only those NT Groups that map to a UNIX group that has a value other than the default (-1) will be exposed in group selection lists in tools that access domain users and groups.

WARNING



The *domain admin group* parameter has been removed in Samba-3 and should no longer be specified in `smb.conf`. This parameter was used to give the listed users membership in the Domain Admins Windows group which gave local admin rights on their workstations (in default configurations).

11.1 Features and Benefits

Samba allows the administrator to create MS Windows NT4/200x group accounts and to arbitrarily associate them with UNIX/Linux group accounts.

Group accounts can be managed using the MS Windows NT4 or MS Windows 200x/XP Professional MMC tools. Appropriate interface scripts should be provided in `smb.conf` if it is desired that UNIX/Linux system accounts should be automatically created when these tools are used. In the absence of these scripts, and so long as **winbindd** is running, Samba group accounts that are created using these tools will be allocated UNIX UIDs/GIDs from the ID range specified by the *idmap uid/idmap gid* parameters in the `smb.conf` file.

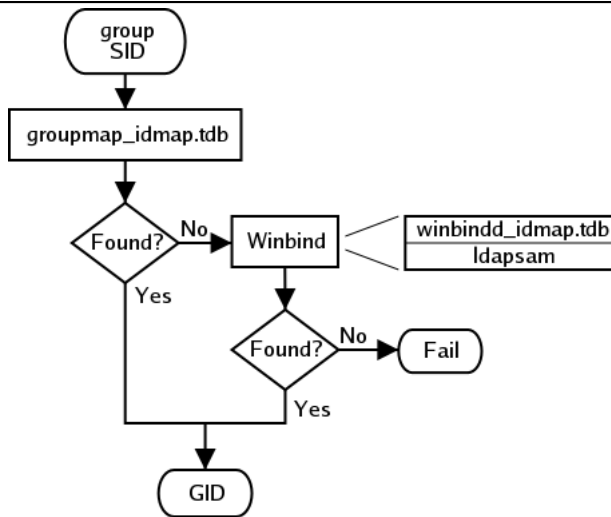


Figure 11.1. IDMAP: group SID to GID resolution.

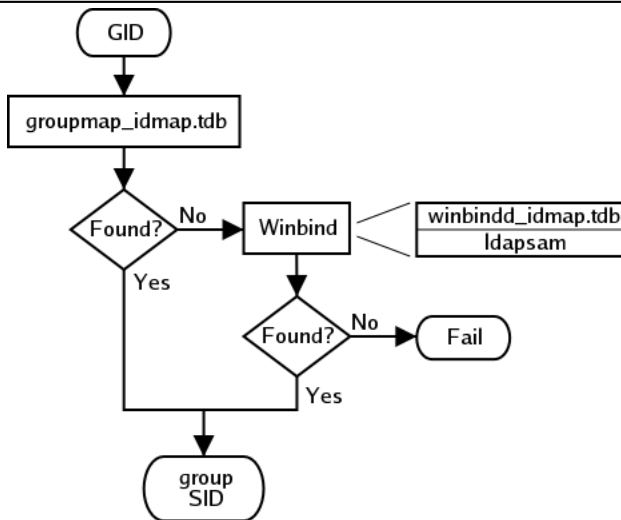


Figure 11.2. IDMAP: GID resolution to matching SID.

In both cases, when winbindd is not running, only locally resolvable groups can be recognized. Please refer to Figure 11.1 and Figure 11.2. The **net groupmap** is used to establish UNIX group to NT SID mappings as shown in Figure 11.3.

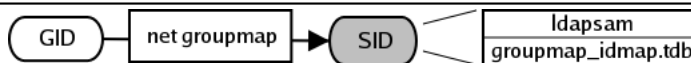


Figure 11.3. IDMAP storing group mappings.

Administrators should be aware that where `smb.conf` group interface scripts make direct calls to the UNIX/Linux system tools (the shadow utilities, `groupadd`, `groupdel`, and `groupmod`), the resulting UNIX/Linux group names will be subject to any limits imposed by these tools. If the tool does not allow upper case characters or space characters, then the creation of an MS Windows NT4/200x style group of *Engineering Managers* will attempt to create an identically named UNIX/Linux group, an attempt that will of course fail.

There are several possible work-arounds for the operating system tools limitation. One method is to use a script that generates a name for the UNIX/Linux system group that fits the operating system limits, and that then just passes the UNIX/Linux group ID (GID) back to the calling Samba interface. This will provide a dynamic work-around solution.

Another work-around is to manually create a UNIX/Linux group, then manually create the MS Windows NT4/200x group on the Samba server and then use the `net groupmap` tool to connect the two to each other.

11.2 Discussion

When installing MS Windows NT4/200x on a computer, the installation program creates default users and groups, notably the `Administrators` group, and gives that group privileges necessary to perform essential system tasks, such as the ability to change the date and time or to kill (or close) any process running on the local machine.

The `Administrator` user is a member of the `Administrators` group, and thus inherits `Administrators` group privileges. If a `joe` user is created to be a member of the `Administrators` group, `joe` has exactly the same rights as the user, `Administrator`.

When an MS Windows NT4/200x/XP machine is made a Domain Member, the “*Domain Admins*” group of the PDC is added to the local `Administrators` group of the workstation. Every member of the `Domain Administrators` group inherits the rights of the local `Administrators` group when logging on the workstation.

The following steps describe how to make Samba PDC users members of the `Domain Admins` group?

1. Create a UNIX group (usually in `/etc/group`), let’s call it `domadm`.
2. Add to this group the users that must be “*Administrators*”. For example, if you want `joe`, `john` and `mary` to be administrators, your entry in `/etc/group` will look like this:

```
domadm:x:502:joe,john,mary
```

3. Map this `domadm` group to the “*Domain Admins*” group by running the command:

```
root# net groupmap add ntgroup=Domain Admins UNIXgroup=domadm
```

The quotes around “*Domain Admins*” are necessary due to the space in the group name. Also make sure to leave no white-space surrounding the equal character (=).

Now joe, john and mary are domain administrators.

It is possible to map any arbitrary UNIX group to any Windows NT4/200x group as well as making any UNIX group a Windows domain group. For example, if you wanted to include a UNIX group (e.g., acct) in an ACL on a local file or printer on a Domain Member machine, you would flag that group as a domain group by running the following on the Samba PDC:

```
root# net groupmap add rid=1000 ntgroup="Accounting" UNIXgroup=acct
```

Be aware that the RID parameter is a unsigned 32-bit integer that should normally start at 1000. However, this RID must not overlap with any RID assigned to a user. Verification for this is done differently depending on the passdb backend you are using. Future versions of the tools may perform the verification automatically, but for now the burden is on you.

11.2.1 Default Users, Groups and Relative Identifiers

When first installed, Microsoft Windows NT4/200x/XP are preconfigured with certain User, Group, and Alias entities. Each has a well-known Relative Identifier (RID). These must be preserved for continued integrity of operation. Samba must be provisioned with certain essential Domain Groups that require the appropriate RID value. When Samba-3 is configured to use `tdbsam` the essential Domain Groups are automatically created. It is the LDAP administrators' responsibility to create (provision) the default NT Groups.

Each essential Domain Group must be assigned its respective well-known RID. The default Users, Groups, Aliases, and RIDs are shown in Table 11.1.

NOTE



When the *passdb backend* uses LDAP (`ldapsam`) it is the administrators' responsibility to create the essential Domain Groups, and to assign each its default RID.

It is permissible to create any Domain Group that may be necessary, just make certain that the essential Domain Groups (well known) have been created and assigned its default RID. Other groups you create may be assigned any arbitrary RID you care to use.

Be sure to map each Domain Group to a UNIX system group. That is the only way to ensure that the group will be available for use as an NT Domain Group.

11.2.2 Example Configuration

You can list the various groups in the mapping database by executing `net groupmap list`. Here is an example:

```
root# net groupmap list
```

Table 11.1. Well-Known User Default RIDs

Well-Known Entity	RID	Type	Essential
Domain Administrator	500	User	No
Domain Guest	501	User	No
Domain KRBTGT	502	User	No
Domain Admins	512	Group	Yes
Domain Users	513	Group	Yes
Domain Guests	514	Group	Yes
Domain Computers	515	Group	No
Domain Controllers	516	Group	No
Domain Certificate Admins	517	Group	No
Domain Schema Admins	518	Group	No
Domain Enterprise Admins	519	Group	No
Domain Policy Admins	520	Group	No
Builtin Admins	544	Alias	No
Builtin users	545	Alias	No
Builtin Guests	546	Alias	No
Builtin Power Users	547	Alias	No
Builtin Account Operators	548	Alias	No
Builtin System Operators	549	Alias	No
Builtin Print Operators	550	Alias	No
Builtin Backup Operators	551	Alias	No
Builtin Replicator	552	Alias	No
Builtin RAS Servers	553	Alias	No

Domain Admins (S-1-5-21-2547222302-1596225915-2414751004-512) -> domadmin

Domain Users (S-1-5-21-2547222302-1596225915-2414751004-513) -> domuser

Domain Guests (S-1-5-21-2547222302-1596225915-2414751004-514) -> domguest

For complete details on **net groupmap**, refer to the net(8) man page.

11.3 Configuration Scripts

Everyone needs tools. Some of us like to create our own, others prefer to use canned tools (i.e., prepared by someone else for general use).

11.3.1 Sample smb.conf Add Group Script

A script to create complying group names for use by the Samba group interfaces is provided in Example 11.1.

The **smb.conf** entry for the above script would be something like that in Example 11.2.

Example 11.1.

```
#!/bin/bash

# Add the group using normal system groupadd tool.
groupadd smbtmpgrp00

thegid='cat /etc/group | grep smbtmpgrp00 | cut -d ":" -f3'

# Now change the name to what we want for the MS Windows networking end
cp /etc/group /etc/group.bak
cat /etc/group.bak | sed s/smbtmpgrp00/$1/g > /etc/group

# Now return the GID as would normally happen.
echo $thegid
exit 0
```

Example 11.2. Configuration of `smb.conf` for the add group script.

```
[global]
...
add_group_script = /path_to_tool/smbgrpadd.sh %g
...
```

11.3.2 Script to Configure Group Mapping

In our example we have created a UNIX/Linux group called *ntadmin*. Our script will create the additional groups *Orks*, *Elves*, and *Gnomes*. It is a good idea to save this shell script for later re-use just in case you ever need to rebuild your mapping database. For the sake of convenience we elect to save this script as a file called `initGroups.sh`. This script is given in Example 11.3.

Of course it is expected that the administrator will modify this to suit local needs. For information regarding the use of the **net groupmap** tool please refer to the man page.

11.4 Common Errors

At this time there are many little surprises for the unwary administrator. In a real sense it is imperative that every step of automated control scripts must be carefully tested manually before putting them into active service.

Example 11.3.

```
#!/bin/bash

net groupmap modify ntgroup="Domain Admins" unixgroup=ntadmin
net groupmap modify ntgroup="Domain Users" unixgroup=users
net groupmap modify ntgroup="Domain Guests" unixgroup=nobody

groupadd Orks
groupadd Elves
groupadd Gnomes

net groupmap add ntgroup="Orks"    unixgroup=Orks    type=d
net groupmap add ntgroup="Elves"  unixgroup=Elves  type=d
net groupmap add ntgroup="Gnomes" unixgroup=Gnomes type=d
```

11.4.1 Adding Groups Fails

This is a common problem when the **groupadd** is called directly by the Samba interface script for the *add group script* in the `smb.conf` file.

The most common cause of failure is an attempt to add an MS Windows group account that has either an upper case character and/or a space character in it.

There are three possible work-arounds. First, use only group names that comply with the limitations of the UNIX/Linux **groupadd** system tool. Second, it involves the use of the script mentioned earlier in this chapter, and third is the option is to manually create a UNIX/Linux group account that can substitute for the MS Windows group name, then use the procedure listed above to map that group to the MS Windows group.

11.4.2 Adding MS Windows Groups to MS Windows Groups Fails

Samba-3 does not support nested groups from the MS Windows control environment.

11.4.3 Adding *Domain Users* to the *Power Users* Group

“*What must I do to add Domain Users to the Power Users group?*” The Power Users group is a group that is local to each Windows 200x/XP Professional workstation. You cannot add the Domain Users group to the Power Users group automatically, it must be done on each workstation by logging in as the local workstation *administrator* and then using the following procedure:

1. Click **Start -> Control Panel -> Users and Passwords**.
2. Click the **Advanced** tab.
3. Click the **Advanced** button.

4. Click **Groups**.
5. Double click **Power Users**. This will launch the panel to add users or groups to the local machine **Power Uses** group.
6. Click the **Add** button.
7. Select the domain from which the **Domain Users** group is to be added.
8. Double click the **Domain Users** group.
9. Click the **Ok** button. If a logon box is presented during this process please remember to enter the connect as `DOMAIN\UserName`. i.e., For the domain `MIDEARTH` and the user `root` enter `MIDEARTH\root`.

FILE, DIRECTORY AND SHARE ACCESS CONTROLS

Advanced MS Windows users are frequently perplexed when file, directory and share manipulation of resources shared via Samba do not behave in the manner they might expect. MS Windows network administrators are often confused regarding network access controls and how to provide users with the access they need while protecting resources from unauthorized access.

Many UNIX administrators are unfamiliar with the MS Windows environment and in particular have difficulty in visualizing what the MS Windows user wishes to achieve in attempts to set file and directory access permissions.

The problem lies in the differences in how file and directory permissions and controls work between the two environments. This difference is one that Samba cannot completely hide, even though it does try to bridge the chasm to a degree.

POSIX Access Control List technology has been available (along with Extended Attributes) for UNIX for many years, yet there is little evidence today of any significant use. This explains to some extent the slow adoption of ACLs into commercial Linux products. MS Windows administrators are astounded at this, given that ACLs were a foundational capability of the now decade-old MS Windows NT operating system.

The purpose of this chapter is to present each of the points of control that are possible with Samba-3 in the hope that this will help the network administrator to find the optimum method for delivering the best environment for MS Windows desktop users.

This is an opportune point to mention that Samba was created to provide a means of interoperability and interchange of data between differing operating environments. Samba has no intent to change UNIX/Linux into a platform like MS Windows. Instead the purpose was and is to provide a sufficient level of exchange of data between the two environments. What is available today extends well beyond early plans and expectations, yet the gap continues to shrink.

12.1 Features and Benefits

Samba offers a lot of flexibility in file system access management. These are the key access control facilities present in Samba today:

SAMBA ACCESS CONTROL FACILITIES

- *UNIX File and Directory Permissions*

Samba honors and implements UNIX file system access controls. Users who access a Samba server will do so as a particular MS Windows user. This information is passed to the Samba server as part of the logon or connection setup process. Samba uses this user identity to validate whether or not the user should be given access to file system resources (files and directories). This chapter provides an overview for those to whom the UNIX permissions and controls are a little strange or unknown.

- *Samba Share Definitions*

In configuring share settings and controls in the `smb.conf` file, the network administrator can exercise overrides to native file system permissions and behaviors. This can be handy and convenient to effect behavior that is more like what MS Windows NT users expect but it is seldom the *best* way to achieve this. The basic options and techniques are described herein.

- *Samba Share ACLs*

Just like it is possible in MS Windows NT to set ACLs on shares themselves, so it is possible to do this in Samba. Few people make use of this facility, yet it remains one of the easiest ways to affect access controls (restrictions) and can often do so with minimum invasiveness compared with other methods.

- *MS Windows ACLs through UNIX POSIX ACLs*

The use of POSIX ACLs on UNIX/Linux is possible only if the underlying operating system supports them. If not, then this option will not be available to you. Current UNIX technology platforms have native support for POSIX ACLs. There are patches for the Linux kernel that also provide this. Sadly, few Linux platforms ship today with native ACLs and Extended Attributes enabled. This chapter has pertinent information for users of platforms that support them.

12.2 File System Access Controls

Perhaps the most important recognition to be made is the simple fact that MS Windows NT4/200x/XP implement a totally divergent file system technology from what is provided in the UNIX operating system environment. First we consider what the most significant differences are, then we look at how Samba helps to bridge the differences.

12.2.1 MS Windows NTFS Comparison with UNIX File Systems

Samba operates on top of the UNIX file system. This means it is subject to UNIX file system conventions and permissions. It also means that if the MS Windows networking

environment requires file system behavior that differs from UNIX file system behavior then somehow Samba is responsible for emulating that in a transparent and consistent manner.

It is good news that Samba does this to a large extent and on top of that provides a high degree of optional configuration to override the default behavior. We look at some of these over-rides, but for the greater part we will stay within the bounds of default behavior. Those wishing to explore the depths of control ability should review the `smb.conf` man page.

The following compares file system features for UNIX with those of Microsoft Windows NT/200x:

Name Space — MS Windows NT4/200x/XP files names may be up to 254 characters long, and UNIX file names may be 1023 characters long. In MS Windows, file extensions indicate particular file types, in UNIX this is not so rigorously observed as all names are considered arbitrary.

What MS Windows calls a folder, UNIX calls a directory.

Case Sensitivity — MS Windows file names are generally upper case if made up of 8.3 (8 character file name and 3 character extension. File names that are longer than 8.3 are case preserving and case insensitive.

UNIX file and directory names are case sensitive and case preserving. Samba implements the MS Windows file name behavior, but it does so as a user application. The UNIX file system provides no mechanism to perform case insensitive file name lookups. MS Windows does this by default. This means that Samba has to carry the processing overhead to provide features that are not native to the UNIX operating system environment.

Consider the following. All are unique UNIX names but one single MS Windows file name:

```
MYFILE.TXT
MyFile.txt
myfile.txt
```

So clearly, in an MS Windows file name space these three files cannot co-exist, but in UNIX they can.

So what should Samba do if all three are present? That which is lexically first will be accessible to MS Windows users, the others are invisible and unaccessible — any other solution would be suicidal.

Directory Separators — MS Windows and DOS uses the backslash `\` as a directory delimiter, and UNIX uses the forward-slash `/` as its directory delimiter. This is handled transparently by Samba.

Drive Identification — MS Windows products support a notion of drive letters, like **C:** to represent disk partitions. UNIX has no concept of separate identifiers for file partitions, each such file system is mounted to become part of the overall directory tree. The UNIX directory tree begins at `/` just like the root of a DOS drive is specified as `C:\`.

File Naming Conventions — MS Windows generally never experiences file names that begin with a dot (`.`) while in UNIX these are commonly found in a user's home directory. Files that begin with a dot (`.`) are typically either start-up files for various UNIX applications, or they may be files that contain start-up configuration data.

Links and Short-Cuts — MS Windows make use of “*links and short-cuts*” that are actually special types of files that will redirect an attempt to execute the file to the real location of the file. UNIX knows of file and directory links, but they are entirely different from what MS Windows users are used to.

Symbolic links are files in UNIX that contain the actual location of the data (file or directory). An operation (like read or write) will operate directly on the file referenced. Symbolic links are also referred to as “*soft links*.” A hard link is something that MS Windows is not familiar with. It allows one physical file to be known simultaneously by more than one file name.

There are many other subtle differences that may cause the MS Windows administrator some temporary discomfort in the process of becoming familiar with UNIX/Linux. These are best left for a text that is dedicated to the purpose of UNIX/Linux training and education.

12.2.2 Managing Directories

There are three basic operations for managing directories: **create**, **delete**, **rename**.

Table 12.1. Managing Directories with UNIX and Windows

Action	MS Windows Command	UNIX Command
create	md folder	mkdir folder
delete	rd folder	rmdir folder
rename	rename oldname newname	mv oldname newname

12.2.3 File and Directory Access Control

The network administrator is strongly advised to read foundational training manuals and reference materials regarding file and directory permissions maintenance. Much can be achieved with the basic UNIX permissions without having to resort to more complex facilities like POSIX Access Control Lists (ACLs) or Extended Attributes (EAs).

UNIX/Linux file and directory access permissions involves setting three primary sets of data and one control set. A UNIX file listing looks as follows:

```

$ ls -la
total 632
drwxr-xr-x  13 maryo  gnomes      816 2003-05-12 22:56 .
drwxrwxr-x  37 maryo  gnomes     3800 2003-05-12 22:29 ..
dr-xr-xr-x   2 maryo  gnomes      48 2003-05-12 22:29 mucho02
drwxrwxrwx   2 maryo  gnomes      48 2003-05-12 22:29 mucho03
drw-rw-rw-   2 maryo  gnomes      48 2003-05-12 22:29 mucho04
d-w--w--w-   2 maryo  gnomes      48 2003-05-12 22:29 mucho05
dr--r--r--   2 maryo  gnomes      48 2003-05-12 22:29 mucho06
drwsrwsrwx   2 maryo  gnomes      48 2003-05-12 22:29 mucho08
-----      1 maryo  gnomes     1242 2003-05-12 22:31 mydata00.lst
--w--w--w-   1 maryo  gnomes     7754 2003-05-12 22:33 mydata02.lst
-r--r--r--   1 maryo  gnomes    21017 2003-05-12 22:32 mydata04.lst
-rw-rw-rw-   1 maryo  gnomes    41105 2003-05-12 22:32 mydata06.lst
$

```

The columns above represent (from left to right): permissions, number of hard links to file, owner, group, size (bytes), access date, access time, file name.

An overview of the permissions field can be found in Figure 12.1.

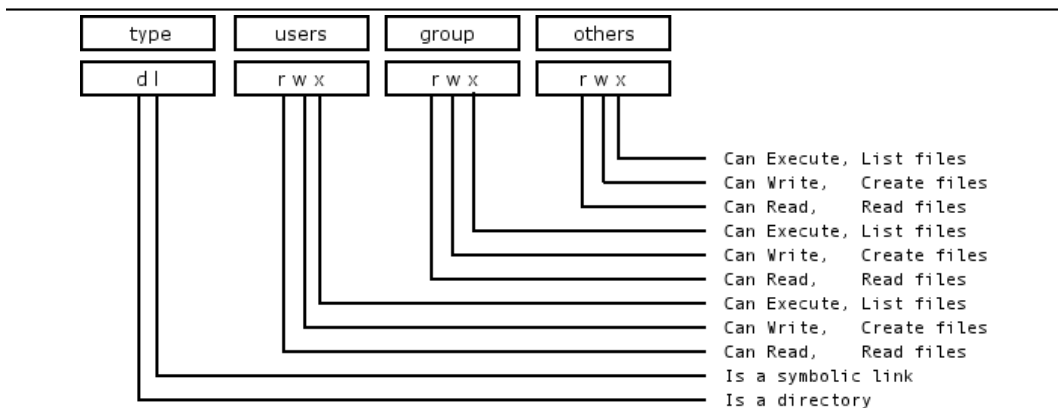


Figure 12.1. Overview of UNIX permissions field.

Any bit flag may be unset. An unset bit flag is the equivalent of “cannot” and is represented as a “-” character.

Example 12.1. Example File

```

-rwxr-x--- Means: The owner (user) can read, write, execute
             the group can read and execute
             everyone else cannot do anything with it.

```

Additional possibilities in the [type] field are: c = character device, b = block device, p = pipe device, s = UNIX Domain Socket.

The letters **rwXst** set permissions for the user, group and others as: read (r), write (w), execute (or access for directories) (x), execute only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution (s), sticky (t).

When the sticky bit is set on a directory, files in that directory may be unlinked (deleted) or renamed only by root or their owner. Without the sticky bit, anyone able to write to the directory can delete or rename files. The sticky bit is commonly found on directories, such as `/tmp`, that are world-writable.

When the set user or group ID bit (s) is set on a directory, then all files created within it will be owned by the user and/or group whose 'set user or group' bit is set. This can be helpful in setting up directories for which it is desired that all users who are in a group should be able to write to and read from a file, particularly when it is undesirable for that file to be exclusively owned by a user whose primary group is not the group that all such users belong to.

When a directory is set **drw-r-----** this means that the owner can read and create (write) files in it, but because the (x) execute flags are not set, files cannot be listed (seen) in the directory by anyone. The group can read files in the directory but cannot create new files. If files in the directory are set to be readable and writable for the group, then group members will be able to write to (or delete) them.

12.3 Share Definition Access Controls

The following parameters in the `smb.conf` file sections define a share control or effect access controls. Before using any of the following options, please refer to the man page for `smb.conf`.

12.3.1 User and Group-Based Controls

User and group-based controls can prove quite useful. In some situations it is distinctly desirable to affect all file system operations as if a single user were doing so. The use of the *force user* and *force group* behavior will achieve this. In other situations it may be necessary to effect a paranoia level of control to ensure that only particular authorized persons will be able to access a share or its contents. Here the use of the *valid users* or the *invalid users* may be most useful.

As always, it is highly advisable to use the least difficult to maintain and the least ambiguous method for controlling access. Remember, when you leave the scene someone else will need to provide assistance and if he finds too great a mess or does not understand what you have done, there is risk of Samba being removed and an alternative solution being adopted.

Table 12.2 enumerates these controls.

Table 12.2. User and Group Based Controls

Control Parameter	Description - Action - Notes
<i>admin users</i>	List of users who will be granted administrative privileges on the share. They will do all file operations as the super-user (root). Any user in this list will be able to do anything they like on the share, irrespective of file permissions.
<i>force group</i>	Specifies a UNIX group name that will be assigned as the default primary group for all users connecting to this service.
<i>force user</i>	Specifies a UNIX user name that will be assigned as the default user for all users connecting to this service. This is useful for sharing files. Incorrect use can cause security problems.
<i>guest ok</i>	If this parameter is set for a service, then no password is required to connect to the service. Privileges will be those of the guest account.
<i>invalid users</i>	List of users that should not be allowed to login to this service.
<i>only user</i>	Controls whether connections with usernames not in the user list will be allowed.
<i>read list</i>	List of users that are given read-only access to a service. Users in this list will not be given write access, no matter what the read only option is set to.
<i>username</i>	Refer to the <code>smb.conf</code> man page for more information – this is a complex and potentially misused parameter.
<i>valid users</i>	List of users that should be allowed to login to this service.
<i>write list</i>	List of users that are given read-write access to a service.

12.3.2 File and Directory Permissions-Based Controls

The following file and directory permission-based controls, if misused, can result in considerable difficulty to diagnose causes of misconfiguration. Use them sparingly and carefully. By gradually introducing each one by one, undesirable side effects may be detected. In the event of a problem, always comment all of them out and then gradually reintroduce them in a controlled way.

Refer to Table 12.3 for information regarding the parameters that may be used to affect file and directory permission-based access controls.

12.3.3 Miscellaneous Controls

The following are documented because of the prevalence of administrators creating inadvertent barriers to file access by not understanding the full implications of `smb.conf` file settings. See Table 12.4.

12.4 Access Controls on Shares

This section deals with how to configure Samba per share access control restrictions. By default, Samba sets no restrictions on the share itself. Restrictions on the share itself can

Table 12.3. File and Directory Permission Based Controls

Control Parameter	Description - Action - Notes
<i>create mask</i>	Refer to the <code>smb.conf</code> man page.
<i>directory mask</i>	The octal modes used when converting DOS modes to UNIX modes when creating UNIX directories. See also: <code>directory security mask</code> .
<i>dos filemode</i>	Enabling this parameter allows a user who has write access to the file to modify the permissions on it.
<i>force create mode</i>	This parameter specifies a set of UNIX mode bit permissions that will always be set on a file created by Samba.
<i>force directory mode</i>	This parameter specifies a set of UNIX mode bit permissions that will always be set on a directory created by Samba.
<i>force directory security mode</i>	Controls UNIX permission bits modified when a Windows NT client is manipulating UNIX permissions on a directory.
<i>force security mode</i>	Controls UNIX permission bits modified when a Windows NT client manipulates UNIX permissions.
<i>hide unreadable</i>	Prevents clients from seeing the existence of files that cannot be read.
<i>hide unwriteable files</i>	Prevents clients from seeing the existence of files that cannot be written to. Unwriteable directories are shown as usual.
<i>nt acl support</i>	This parameter controls whether <code>smbd</code> will attempt to map UNIX permissions into Windows NT access control lists.
<i>security mask</i>	Controls UNIX permission bits modified when a Windows NT client is manipulating the UNIX permissions on a file.

be set on MS Windows NT4/200x/XP shares. This can be an effective way to limit who can connect to a share. In the absence of specific restrictions the default setting is to allow the global user `Everyone - Full Control` (full control, change and read).

At this time Samba does not provide a tool for configuring access control setting on the share itself. Samba does have the capacity to store and act on access control settings, but the only way to create those settings is to use either the NT4 Server Manager or the Windows 200x MMC for Computer Management.

Samba stores the per share access control settings in a file called `share_info.tdb`. The location of this file on your system will depend on how Samba was compiled. The default location for Samba's tdb files is under `/usr/local/samba/var`. If the `tdbdump` utility has been compiled and installed on your system, then you can examine the contents of this file by executing: `tdbdump share_info.tdb` in the directory containing the tdb files.

Table 12.4. Other Controls

Control Parameter	Description - Action - Notes
<i>case sensitive, default case, short preserve case</i>	This means that all file name lookup will be done in a case sensitive manner. Files will be created with the precise file name Samba received from the MS Windows client.
<i>csc policy</i>	Client Side Caching Policy - parallels MS Windows client side file caching capabilities.
<i>dont descend</i>	Allows specifying a comma-delimited list of directories that the server should always show as empty.
<i>dos filetime resolution</i>	This option is mainly used as a compatibility option for Visual C++ when used against Samba shares.
<i>dos filetimes</i>	DOS and Windows allow users to change file time stamps if they can write to the file. POSIX semantics prevent this. This option allows DOS and Windows behavior.
<i>fake oplocks</i>	Oplocks are the way that SMB clients get permission from a server to locally cache file operations. If a server grants an oplock, the client is free to assume that it is the only one accessing the file and it will aggressively cache file data.
<i>hide dot files, hide files, veto files</i>	Note: MS Windows Explorer allows override of files marked as hidden so they will still be visible.
<i>read only</i>	If this parameter is yes, then users of a service may not create or modify files in the service's directory.
<i>veto files</i>	List of files and directories that are neither visible nor accessible.

12.4.1 Share Permissions Management

The best tool for the task is platform dependant. Choose the best tool for your environment.

12.4.1.1 Windows NT4 Workstation/Server

The tool you need to use to manage share permissions on a Samba server is the NT Server Manager. Server Manager is shipped with Windows NT4 Server products but not with Windows NT4 Workstation. You can obtain the NT Server Manager for MS Windows NT4 Workstation from Microsoft — see details below.

INSTRUCTIONS

1. Launch the NT4 Server Manager, click on the Samba server you want to administer. From the menu select **Computer**, then click on **Shared Directories**.
2. Click on the share that you wish to manage, then click the **Properties** tab. then click the **Permissions** tab. Now you can add or change access control settings as you wish.

12.4.1.2 Windows 200x/XP

On MS Windows NT4/200x/XP system access control lists on the share itself are set using native tools, usually from File Manager. For example, in Windows 200x, right click on the shared folder, then select **Sharing**, then click on **Permissions**. The default Windows NT4/200x permission allows “*Everyone*” full control on the share.

MS Windows 200x and later versions come with a tool called the Computer Management snap-in for the Microsoft Management Console (MMC). This tool is located by clicking on **Control Panel -> Administrative Tools -> Computer Management**.

INSTRUCTIONS

1. After launching the MMC with the Computer Management snap-in, click the menu item **Action**, and select **Connect to another computer**. If you are not logged onto a domain you will be prompted to enter a domain login user identifier and a password. This will authenticate you to the domain. If you are already logged in with administrative privilege, this step is not offered.
2. If the Samba server is not shown in the **Select Computer** box, type in the name of the target Samba server in the field **Name:**. Now click the on **[+]** next to **System Tools**, then on the **[+]** next to **Shared Folders** in the left panel.
3. In the right panel, double-click on the share on which you wish to set access control permissions. Then click the tab **Share Permissions**. It is now possible to add access control entities to the shared folder. Remember to set what type of access (full control, change, read) you wish to assign for each entry.

WARNING

Be careful. If you take away all permissions from the Everyone user without removing this user, effectively no user will be able to access the share. This is a result of what is known as ACL precedence. Everyone with *no access* means that MaryK who is part of the group Everyone will have no access even if she is given explicit full control access.

12.5 MS Windows Access Control Lists and UNIX Interoperability

12.5.1 Managing UNIX Permissions Using NT Security Dialogs

Windows NT clients can use their native security settings dialog box to view and modify the underlying UNIX permissions.

This ability is careful not to compromise the security of the UNIX host on which Samba is running, and still obeys all the file permission rules that a Samba administrator can set.

Samba does not attempt to go beyond POSIX ACLs, so the various finer-grained access control options provided in Windows are actually ignored.

NOTE



All access to UNIX/Linux system files via Samba is controlled by the operating system file access controls. When trying to figure out file access problems, it is vitally important to find the identity of the Windows user as it is presented by Samba at the point of file access. This can best be determined from the Samba log files.

12.5.2 Viewing File Security on a Samba Share

From an NT4/2000/XP client, right click on any file or directory in a Samba-mounted drive letter or UNC path. When the menu pops up, click on the **Properties** entry at the bottom of the menu. This brings up the file **Properties** dialog box. Click on the **Security** tab and you will see three buttons: **Permissions**, **Auditing**, and **Ownership**. The **Auditing** button will cause either an error message ‘A requested privilege is not held by the client’ to appear if the user is not the NT Administrator, or a dialog which is intended to allow an Administrator to add auditing requirements to a file if the user is logged on as the NT Administrator. This dialog is non-functional with a Samba share at this time, as the only useful button, the **Add** button, will not currently allow a list of users to be seen.

12.5.3 Viewing File Ownership

Clicking on the **Ownership** button brings up a dialog box telling you who owns the given file. The owner name will be displayed like this:

“*SERVER\user (Long name)*”

SERVER is the NetBIOS name of the Samba server, *user* is the user name of the UNIX user who owns the file, and *(Long name)* is the descriptive string identifying the user (normally found in the GECOS field of the UNIX password database). Click on the **Close** button to remove this dialog.

If the parameter *nt acl support* is set to **false**, the file owner will be shown as the NT user *Everyone*.

The **Take Ownership** button will not allow you to change the ownership of this file to yourself (clicking it will display a dialog box complaining that the user you are currently logged onto the NT client cannot be found). The reason for this is that changing the ownership of a file is a privileged operation in UNIX, available only to the *root* user. As clicking on this button causes NT to attempt to change the ownership of a file to the current user logged into the NT client, this will not work with Samba at this time.

There is an NT **chown** command that will work with Samba and allow a user with Administrator privilege connected to a Samba server as root to change the ownership of files on both a local NTFS filesystem or remote mounted NTFS or Samba drive. This is available as part of the Seclib NT security library written by Jeremy Allison of the Samba Team, and is available from the main Samba FTP site.

12.5.4 Viewing File or Directory Permissions

The third button is the **Permissions** button. Clicking on this brings up a dialog box that shows both the permissions and the UNIX owner of the file or directory. The owner is displayed like this:

```
SERVER\user (Long name)
```

Where *SERVER* is the NetBIOS name of the Samba server, *user* is the user name of the UNIX user who owns the file, and (*Long name*) is the descriptive string identifying the user (normally found in the GECOS field of the UNIX password database).

If the parameter *nt acl support* is set to **false**, the file owner will be shown as the NT user **Everyone** and the permissions will be shown as NT “*Full Control*”.

The permissions field is displayed differently for files and directories, so I’ll describe the way file permissions are displayed first.

12.5.4.1 File Permissions

The standard UNIX user/group/world triplet and the corresponding **read**, **write**, **execute** permissions triplets are mapped by Samba into a three element NT ACL with the “*r*”, “*w*” and “*x*” bits mapped into the corresponding NT permissions. The UNIX world permissions are mapped into the global NT group **Everyone**, followed by the list of permissions allowed for UNIX world. The UNIX owner and group permissions are displayed as an NT **user** icon and an NT **local group** icon, respectively, followed by the list of permissions allowed for the UNIX user and group.

Because many UNIX permission sets do not map into common NT names such as **read**, **change** or **full control**, usually the permissions will be prefixed by the words **Special Access** in the NT display list.

But what happens if the file has no permissions allowed for a particular UNIX user group or world component? In order to allow “*no permissions*” to be seen and modified Samba then overloads the NT **Take Ownership** ACL attribute (which has no meaning in UNIX) and reports a component with no permissions as having the NT **O** bit set. This was chosen, of course, to make it look like a zero, meaning zero permissions. More details on the decision behind this is given below.

12.5.4.2 Directory Permissions

Directories on an NT NTFS file system have two different sets of permissions. The first set is the ACL set on the directory itself, which is usually displayed in the first set of parentheses in the normal RW NT style. This first set of permissions is created by Samba in exactly the same way as normal file permissions are, described above, and is displayed in the same way.

The second set of directory permissions has no real meaning in the UNIX permissions world and represents the *inherited* permissions that any file created within this directory would inherit.

Samba synthesises these inherited permissions for NT by returning as an NT ACL the UNIX permission mode that a new file created by Samba on this share would receive.

12.5.5 Modifying File or Directory Permissions

Modifying file and directory permissions is as simple as changing the displayed permissions in the dialog box, and clicking on **OK**. However, there are limitations that a user needs to be aware of, and also interactions with the standard Samba permission masks and mapping of DOS attributes that need to also be taken into account.

If the parameter *nt acl support* is set to **false**, any attempt to set security permissions will fail with an ‘Access Denied’ message.

The first thing to note is that the **Add** button will not return a list of users in Samba (it will give an error message saying ‘The remote procedure call failed and did not execute’). This means that you can only manipulate the current user/group/world permissions listed in the dialog box. This actually works quite well as these are the only permissions that UNIX actually has.

If a permission triplet (either user, group, or world) is removed from the list of permissions in the NT dialog box, then when the **OK** button is pressed it will be applied as “*no permissions*” on the UNIX side. If you then view the permissions again, the “*no permissions*” entry will appear as the NT **O** flag, as described above. This allows you to add permissions back to a file or directory once you have removed them from a triplet component.

As UNIX supports only the “*r*”, “*w*” and “*x*” bits of an NT ACL, if other NT security attributes such as **Delete Access** are selected they will be ignored when applied on the Samba server.

When setting permissions on a directory, the second set of permissions (in the second set of parentheses) is by default applied to all files within that directory. If this is not what you want, you must uncheck the **Replace permissions on existing files** checkbox in the NT dialog before clicking on **OK**.

If you wish to remove all permissions from a user/group/world component, you may either highlight the component and click on the **Remove** button, or set the component to only have the special **Take Ownership** permission (displayed as **O**) highlighted.

12.5.6 Interaction with the Standard Samba “*create mask*” Parameters

There are four parameters that control interaction with the standard Samba *create mask* parameters. These are:

- *security mask*
- *force security mode*
- *directory security mask*
- *force directory security mode*

Once a user clicks on **OK** to apply the permissions, Samba maps the given permissions into a user/group/world r/w/x triplet set, and then checks the changed permissions for a file against the bits set in the *security mask* parameter. Any bits that were changed that are not set to “1” in this parameter are left alone in the file permissions.

Essentially, zero bits in the *security mask* may be treated as a set of bits the user is *not* allowed to change, and one bits are those the user is allowed to change.

If not explicitly set, this parameter defaults to the same value as the *create mask* parameter. To allow a user to modify all the user/group/world permissions on a file, set this parameter to 0777.

Next Samba checks the changed permissions for a file against the bits set in the *force security mode* parameter. Any bits that were changed that correspond to bits set to “1” in this parameter are forced to be set.

Essentially, bits set in the *force security mode* parameter may be treated as a set of bits that, when modifying security on a file, the user has always set to be “on”.

If not explicitly set, this parameter defaults to the same value as the *force create mode* parameter. To allow a user to modify all the user/group/world permissions on a file with no restrictions set this parameter to 000. The *security mask* and *force security mode* parameters are applied to the change request in that order.

For a directory, Samba will perform the same operations as described above for a file except it uses the parameter *directory security mask* instead of *security mask*, and *force directory security mode* parameter instead of *force security mode*.

The *directory security mask* parameter by default is set to the same value as the *directory mask* parameter and the *force directory security mode* parameter by default is set to the same value as the *force directory mode* parameter. In this way Samba enforces the permission restrictions that an administrator can set on a Samba share, while still allowing users to modify the permission bits within that restriction.

If you want to set up a share that allows users full control in modifying the permission bits on their files and directories and does not force any particular bits to be set “on”, then set the following parameters in the `smb.conf` file in that share-specific section:

```
security mask = 0777
force security mode = 0
directory security mask = 0777
force directory security mode = 0
```


12.5.7 Interaction with the Standard Samba File Attribute Mapping

NOTE



Samba maps some of the DOS attribute bits (such as “*read only*”) into the UNIX permissions of a file. This means there can be a conflict between the permission bits set via the security dialog and the permission bits set by the file attribute mapping.

If a file has no UNIX read access for the owner, it will show up as “*read only*” in the standard file attributes tabbed dialog. Unfortunately, this dialog is the same one that contains the security information in another tab.

What this can mean is that if the owner changes the permissions to allow himself read access using the security dialog, clicks on **OK** to get back to the standard attributes tab dialog, and clicks on **OK** on that dialog, then NT will set the file permissions back to read-only (as that is what the attributes still say in the dialog). This means that after setting permissions and clicking on **OK** to get back to the attributes dialog, you should always press **Cancel** rather than **OK** to ensure that your changes are not overridden.

12.6 Common Errors

File, directory and share access problems are common on the mailing list. The following are examples taken from the mailing list in recent times.

12.6.1 Users Cannot Write to a Public Share

*“We are facing some troubles with file/directory permissions. I can log on the domain as admin user(root), and there’s a public share on which everyone needs to have permission to create/modify files, but only root can change the file, no one else can. We need to constantly go to the server to `chgrp -R users *` and `chown -R nobody *` to allow others users to change the file.”*

There are many ways to solve this problem and here are a few hints:

1. Go to the top of the directory that is shared.
2. Set the ownership to what ever public owner and group you want

```
$ find 'directory_name' -type d -exec chown user.group {} \;  
$ find 'directory_name' -type d -exec chmod 6775 'directory_name'  
$ find 'directory_name' -type f -exec chmod 0775 {} \;  
$ find 'directory_name' -type f -exec chown user.group {} \;
```

NOTE



The above will set the sticky bit on all directories. Read your UNIX/Linux man page on what that does. It causes the OS to assign to all files created in the directories the ownership of the directory.

3. Directory is: */foodbar*

```
$ chown jack.engr /foodbar
```

NOTE



This is the same as doing:

```
$ chown jack /foodbar  
$ chgrp engr /foodbar
```

4. Now type:

```
$ chmod 6775 /foodbar  
$ ls -al /foodbar/..
```

You should see:

```
drwsrwsr-x  2 jack  engr    48 2003-02-04 09:55 foodbar
```

5. Now type:

```
$ su - jill  
$ cd /foodbar  
$ touch Afile  
$ ls -al
```

You should see that the file **Afile** created by Jill will have ownership and permissions of Jack, as follows:

```
-rw-r--r--  1 jack  engr      0 2003-02-04 09:57 Afile
```

6. Now in your `smb.conf` for the share add:

```
force create mode = 0775
force directory mode = 6775
```

NOTE



These procedures are needed only if your users are not members of the group you have used. That is if within the OS do not have write permission on the directory.

An alternative is to set in the `smb.conf` entry for the share:

```
force user = jack
force group = engr
```

12.6.2 File Operations Done as *root* with *force user* Set

When you have a user in *admin users*, Samba will always do file operations for this user as *root*, even if *force user* has been set.

12.6.3 MS Word with Samba Changes Owner of File

Question: “When user *B* saves a word document that is owned by user *A* the updated file is now owned by user *B*. Why is Samba doing this? How do I fix this?”

Answer: Word does the following when you modify/change a Word document: MS Word creates a NEW document with a temporary name, Word then closes the old document and deletes it, Word then renames the new document to the original document name. There is no mechanism by which Samba can in any way know that the new document really should be owned by the owners of the original file. Samba has no way of knowing that the file will be renamed by MS Word. As far as Samba is able to tell, the file that gets created is a NEW file, not one that the application (Word) is updating.

There is a work-around to solve the permissions problem. That work-around involves understanding how you can manage file system behavior from within the `smb.conf` file, as well as understanding how UNIX file systems work. Set on the directory in which you are changing Word documents: **chmod g+s ‘directory_name’** This ensures that all files will be created with the group that owns the directory. In `smb.conf` share declaration section set:

```
force create mode = 0660
force directory mode = 0770
```

These two settings will ensure that all directories and files that get created in the share will be read/writable by the owner and group set on the directory itself.

FILE AND RECORD LOCKING

One area that causes trouble for many network administrators is locking. The extent of the problem is readily evident from searches over the Internet.

13.1 Features and Benefits

Samba provides all the same locking semantics that MS Windows clients expect and that MS Windows NT4/200x servers also provide.

The term *locking* has exceptionally broad meaning and covers a range of functions that are all categorized under this one term.

Opportunistic locking is a desirable feature when it can enhance the perceived performance of applications on a networked client. However, the opportunistic locking protocol is not robust and, therefore, can encounter problems when invoked beyond a simplistic configuration or on extended slow or faulty networks. In these cases, operating system management of opportunistic locking and/or recovering from repetitive errors can offset the perceived performance advantage that it is intended to provide.

The MS Windows network administrator needs to be aware that file and record locking semantics (behavior) can be controlled either in Samba or by way of registry settings on the MS Windows client.

NOTE



Sometimes it is necessary to disable locking control settings on both the Samba server as well as on each MS Windows client!

13.2 Discussion

There are two types of locking that need to be performed by an SMB server. The first is *record locking* that allows a client to lock a range of bytes in an open file. The second is the

deny modes that are specified when a file is open.

Record locking semantics under UNIX are very different from record locking under Windows. Versions of Samba before 2.2 have tried to use the native `fcntl()` UNIX system call to implement proper record locking between different Samba clients. This cannot be fully correct for several reasons. The simplest is the fact that a Windows client is allowed to lock a byte range up to 2^{32} or 2^{64} , depending on the client OS. The UNIX locking only supports byte ranges up to 2^{31} . So it is not possible to correctly satisfy a lock request above 2^{31} . There are many more differences, too many to be listed here.

Samba 2.2 and above implements record locking completely independent of the underlying UNIX system. If a byte range lock that the client requests happens to fall into the range of $0-2^{31}$, Samba hands this request down to the UNIX system. All other locks cannot be seen by UNIX, anyway.

Strictly speaking, an SMB server should check for locks before every read and write call on a file. Unfortunately with the way `fcntl()` works, this can be slow and may overstress the **rpc.lockd**. This is almost always unnecessary as clients are supposed to independently make locking calls before reads and writes if locking is important to them. By default, Samba only makes locking calls when explicitly asked to by a client, but if you set *strict locking* = yes, it will make lock checking calls on *every* read and write call.

You can also disable byte range locking completely by using *locking* = no. This is useful for those shares that do not support locking or do not need it (such as CDROMs). In this case, Samba fakes the return codes of locking calls to tell clients that everything is okay.

The second class of locking is the *deny modes*. These are set by an application when it opens a file to determine what types of access should be allowed simultaneously with its open. A client may ask for `DENY_NONE`, `DENY_READ`, `DENY_WRITE`, or `DENY_ALL`. There are also special compatibility modes called `DENY_FCB` and `DENY_DOS`.

13.2.1 Opportunistic Locking Overview

Opportunistic locking (Oplocks) is invoked by the Windows file system (as opposed to an API) via registry entries (on the server and the client) for the purpose of enhancing network performance when accessing a file residing on a server. Performance is enhanced by caching the file locally on the client that allows:

Read-ahead: — The client reads the local copy of the file, eliminating network latency.

Write caching: — The client writes to the local copy of the file, eliminating network latency.

Lock caching: — The client caches application locks locally, eliminating network latency.

The performance enhancement of oplocks is due to the opportunity of exclusive access to the file — even if it is opened with `deny-none` — because Windows monitors the file's status for concurrent access from other processes.

WINDOWS DEFINES 4 KINDS OF OPLOCKS:

Level1 Oplock — The redirector sees that the file was opened with deny none (allowing concurrent access), verifies that no other process is accessing the file, checks that oplocks are enabled, then grants deny-all/read-write/exclusive access to the file. The client now performs operations on the cached local file.

If a second process attempts to open the file, the open is deferred while the redirector “*breaks*” the original oplock. The oplock break signals the caching client to write the local file back to the server, flush the local locks and discard read-ahead data. The break is then complete, the deferred open is granted, and the multiple processes can enjoy concurrent file access as dictated by mandatory or byte-range locking options. However, if the original opening process opened the file with a share mode other than deny-none, then the second process is granted limited or no access, despite the oplock break.

Level2 Oplock — Performs like a Level1 oplock, except caching is only operative for reads. All other operations are performed on the server disk copy of the file.

Filter Oplock — Does not allow write or delete file access.

Batch Oplock — Manipulates file openings and closings and allows caching of file attributes.

An important detail is that oplocks are invoked by the file system, not an application API. Therefore, an application can close an oplocked file, but the file system does not relinquish the oplock. When the oplock break is issued, the file system then simply closes the file in preparation for the subsequent open by the second process.

Opportunistic locking is actually an improper name for this feature. The true benefit of this feature is client-side data caching, and oplocks is merely a notification mechanism for writing data back to the networked storage disk. The limitation of opportunistic locking is the reliability of the mechanism to process an oplock break (notification) between the server and the caching client. If this exchange is faulty (usually due to timing out for any number of reasons), then the client-side caching benefit is negated.

The actual decision that a user or administrator should consider is whether it is sensible to share among multiple users data that will be cached locally on a client. In many cases the answer is no. Deciding when to cache or not cache data is the real question, and thus “*opportunistic locking*” should be treated as a toggle for client-side caching. Turn it “*on*” when client-side caching is desirable and reliable. Turn it “*off*” when client-side caching is redundant, unreliable or counter-productive.

Opportunistic locking is by default set to “*on*” by Samba on all configured shares, so careful attention should be given to each case to determine if the potential benefit is worth the potential for delays. The following recommendations will help to characterize the environment where opportunistic locking may be effectively configured.

Windows opportunistic locking is a lightweight performance-enhancing feature. It is not

a robust and reliable protocol. Every implementation of opportunistic locking should be evaluated as a tradeoff between perceived performance and reliability. Reliability decreases as each successive rule above is not enforced. Consider a share with oplocks enabled, over a wide area network, to a client on a South Pacific atoll, on a high-availability server, serving a mission-critical multi-user corporate database during a tropical storm. This configuration will likely encounter problems with oplocks.

Oplocks can be beneficial to perceived client performance when treated as a configuration toggle for client-side data caching. If the data caching is likely to be interrupted, then oplock usage should be reviewed. Samba enables opportunistic locking by default on all shares. Careful attention should be given to the client usage of shared data on the server, the server network reliability and the opportunistic locking configuration of each share. In mission critical high availability environments, data integrity is often a priority. Complex and expensive configurations are implemented to ensure that if a client loses connectivity with a file server, a failover replacement will be available immediately to provide continuous data availability.

Windows client failover behavior is more at risk of application interruption than other platforms because it is dependent upon an established TCP transport connection. If the connection is interrupted — as in a file server failover — a new session must be established. It is rare for Windows client applications to be coded to recover correctly from a transport connection loss, therefore, most applications will experience some sort of interruption — at worst, abort and require restarting.

If a client session has been caching writes and reads locally due to opportunistic locking, it is likely that the data will be lost when the application restarts or recovers from the TCP interrupt. When the TCP connection drops, the client state is lost. When the file server recovers, an oplock break is not sent to the client. In this case, the work from the prior session is lost. Observing this scenario with oplocks disabled and with the client writing data to the file server real-time, the failover will provide the data on disk as it existed at the time of the disconnect.

In mission-critical high-availability environments, careful attention should be given to opportunistic locking. Ideally, comprehensive testing should be done with all affected applications with oplocks enabled and disabled.

13.2.1.1 Exclusively Accessed Shares

Opportunistic locking is most effective when it is confined to shares that are exclusively accessed by a single user, or by only one user at a time. Because the true value of opportunistic locking is the local client caching of data, any operation that interrupts the caching mechanism will cause a delay.

Home directories are the most obvious examples of where the performance benefit of opportunistic locking can be safely realized.

13.2.1.2 Multiple-Accessed Shares or Files

As each additional user accesses a file in a share with opportunistic locking enabled, the potential for delays and resulting perceived poor performance increases. When multiple

users are accessing a file on a share that has oplocks enabled, the management impact of sending and receiving oplock breaks and the resulting latency while other clients wait for the caching client to flush data offset the performance gains of the caching user.

As each additional client attempts to access a file with oplocks set, the potential performance improvement is negated and eventually results in a performance bottleneck.

13.2.1.3 UNIX or NFS Client-Accessed Files

Local UNIX and NFS clients access files without a mandatory file-locking mechanism. Thus, these client platforms are incapable of initiating an oplock break request from the server to a Windows client that has a file cached. Local UNIX or NFS file access can therefore write to a file that has been cached by a Windows client, which exposes the file to likely data corruption.

If files are shared between Windows clients, and either local UNIX or NFS users, turn opportunistic locking off.

13.2.1.4 Slow and/or Unreliable Networks

The biggest potential performance improvement for opportunistic locking occurs when the client-side caching of reads and writes delivers the most differential over sending those reads and writes over the wire. This is most likely to occur when the network is extremely slow, congested, or distributed (as in a WAN). However, network latency also has a high impact on the reliability of the oplock break mechanism, and thus increases the likelihood of encountering oplock problems that more than offset the potential perceived performance gain. Of course, if an oplock break never has to be sent, then this is the most advantageous scenario to utilize opportunistic locking.

If the network is slow, unreliable, or a WAN, then do not configure opportunistic locking if there is any chance of multiple users regularly opening the same file.

13.2.1.5 Multi-User Databases

Multi-user databases clearly pose a risk due to their very nature — they are typically heavily accessed by numerous users at random intervals. Placing a multi-user database on a share with opportunistic locking enabled will likely result in a locking management bottleneck on the Samba server. Whether the database application is developed in-house or a commercially available product, ensure that the share has opportunistic locking disabled.

13.2.1.6 PDM Data Shares

Process Data Management (PDM) applications such as IMAN, Enovia and Clearcase are increasing in usage with Windows client platforms, and therefore SMB datastores. PDM applications manage multi-user environments for critical data security and access. The typical PDM environment is usually associated with sophisticated client design applications that will load data locally as demanded. In addition, the PDM application will usually monitor the data-state of each client. In this case, client-side data caching is best left to the

local application and PDM server to negotiate and maintain. It is appropriate to eliminate the client OS from any caching tasks, and the server from any oplock management, by disabling opportunistic locking on the share.

13.2.1.7 Beware of Force User

Samba includes an `smb.conf` parameter called *force user* that changes the user accessing a share from the incoming user to whatever user is defined by the `smb.conf` variable. If opportunistic locking is enabled on a share, the change in user access causes an oplock break to be sent to the client, even if the user has not explicitly loaded a file. In cases where the network is slow or unreliable, an oplock break can become lost without the user even accessing a file. This can cause apparent performance degradation as the client continually reconnects to overcome the lost oplock break.

Avoid the combination of the following:

- *force user* in the `smb.conf` share configuration.
- Slow or unreliable networks
- Opportunistic locking enabled

13.2.1.8 Advanced Samba Opportunistic Locking Parameters

Samba provides opportunistic locking parameters that allow the administrator to adjust various properties of the oplock mechanism to account for timing and usage levels. These parameters provide good versatility for implementing oplocks in environments where they would likely cause problems. The parameters are: *oplock break wait time*, *oplock contention limit*.

For most users, administrators and environments, if these parameters are required, then the better option is to simply turn oplocks off. The Samba SWAT help text for both parameters reads: “*Do not change this parameter unless you have read and understood the Samba oplock code.*” This is good advice.

13.2.1.9 Mission-Critical High-Availability

In mission-critical high-availability environments, data integrity is often a priority. Complex and expensive configurations are implemented to ensure that if a client loses connectivity with a file server, a failover replacement will be available immediately to provide continuous data availability.

Windows client failover behavior is more at risk of application interruption than other platforms because it is dependant upon an established TCP transport connection. If the connection is interrupted — as in a file server failover — a new session must be established. It is rare for Windows client applications to be coded to recover correctly from a transport connection loss, therefore, most applications will experience some sort of interruption — at worst, abort and require restarting.

If a client session has been caching writes and reads locally due to opportunistic locking, it is likely that the data will be lost when the application restarts, or recovers from the TCP interrupt. When the TCP connection drops, the client state is lost. When the file server recovers, an oplock break is not sent to the client. In this case, the work from the prior session is lost. Observing this scenario with oplocks disabled, and the client was writing data to the file server real-time, then the failover will provide the data on disk as it existed at the time of the disconnect.

In mission-critical high-availability environments, careful attention should be given to opportunistic locking. Ideally, comprehensive testing should be done with all effected applications with oplocks enabled and disabled.

13.3 Samba Opportunistic Locking Control

Opportunistic locking is a unique Windows file locking feature. It is not really file locking, but is included in most discussions of Windows file locking, so is considered a de facto locking feature. Opportunistic locking is actually part of the Windows client file caching mechanism. It is not a particularly robust or reliable feature when implemented on the variety of customized networks that exist in enterprise computing.

Like Windows, Samba implements opportunistic locking as a server-side component of the client caching mechanism. Because of the lightweight nature of the Windows feature design, effective configuration of opportunistic locking requires a good understanding of its limitations, and then applying that understanding when configuring data access for each particular customized network and client usage state.

Opportunistic locking essentially means that the client is allowed to download and cache a file on their hard drive while making changes; if a second client wants to access the file, the first client receives a break and must synchronize the file back to the server. This can give significant performance gains in some cases; some programs insist on synchronizing the contents of the entire file back to the server for a single change.

Level1 Oplocks (also known as just plain “*oplocks*”) is another term for opportunistic locking.

Level2 Oplocks provides opportunistic locking for a file that will be treated as *read only*. Typically this is used on files that are read-only or on files that the client has no initial intention to write to at time of opening the file.

Kernel Oplocks are essentially a method that allows the Linux kernel to co-exist with Samba’s oplocked files, although this has provided better integration of MS Windows network file locking with the underlying OS, SGI IRIX and Linux are the only two OSs that are oplock-aware at this time.

Unless your system supports kernel oplocks, you should disable oplocks if you are accessing the same files from both UNIX/Linux and SMB clients. Regardless, oplocks should always be disabled if you are sharing a database file (e.g., Microsoft Access) between multiple clients, as any break the first client receives will affect synchronization of the entire file (not just the single record), which will result in a noticeable performance impairment and, more likely, problems accessing the database in the first place. Notably, Microsoft Outlook’s personal folders (*.pst) react quite badly to oplocks. If in doubt, disable oplocks and tune your system from that point.

If client-side caching is desirable and reliable on your network, you will benefit from turning on oplocks. If your network is slow and/or unreliable, or you are sharing your files among other file sharing mechanisms (e.g., NFS) or across a WAN, or multiple people will be accessing the same files frequently, you probably will not benefit from the overhead of your client sending oplock breaks and will instead want to disable oplocks for the share.

Another factor to consider is the perceived performance of file access. If oplocks provide no measurable speed benefit on your network, it might not be worth the hassle of dealing with them.

13.3.1 Example Configuration

In the following section we examine two distinct aspects of Samba locking controls.

13.3.1.1 Disabling Oplocks

You can disable oplocks on a per-share basis with the following:

```
oplocks = False  
level2 oplocks = False
```

The default oplock type is Level1. Level2 oplocks are enabled on a per-share basis in the `smb.conf` file.

Alternately, you could disable oplocks on a per-file basis within the share:

```
veto oplock files = /*.mdb/*.MDB/*.dbf/*.DBF/
```

If you are experiencing problems with oplocks as apparent from Samba's log entries, you may want to play it safe and disable oplocks and Level2 oplocks.

13.3.1.2 Disabling Kernel Oplocks

Kernel oplocks is an `smb.conf` parameter that notifies Samba (if the UNIX kernel has the capability to send a Windows client an oplock break) when a UNIX process is attempting to open the file that is cached. This parameter addresses sharing files between UNIX and Windows with oplocks enabled on the Samba server: the UNIX process can open the file that is Oplocked (cached) by the Windows client and the `smbd` process will not send an oplock break, which exposes the file to the risk of data corruption. If the UNIX kernel has the ability to send an oplock break, then the kernel oplocks parameter enables Samba to send the oplock break. Kernel oplocks are enabled on a per-server basis in the `smb.conf` file.

```
kernel oplocks = yes
```

The default is no.

Veto opLocks is an `smb.conf` parameter that identifies specific files for which oplocks are disabled. When a Windows client opens a file that has been configured for veto oplocks, the client will not be granted the oplock, and all operations will be executed on the original file on disk instead of a client-cached file copy. By explicitly identifying files that are shared with UNIX processes and disabling oplocks for those files, the server-wide Oplock configuration

can be enabled to allow Windows clients to utilize the performance benefit of file caching without the risk of data corruption. Veto Oplocks can be enabled on a per-share basis, or globally for the entire server, in the `smb.conf` file as shown in Example 13.1.

Example 13.1. Share with some files oplocked

```
[global]
veto oplock files = /filename.htm/*.txt/

[share_name]
veto oplock files = /*.exe/filename.ext/
```

`oplock break wait time` is an `smb.conf` parameter that adjusts the time interval for Samba to reply to an oplock break request. Samba recommends: “*Do not change this parameter unless you have read and understood the Samba oplock code.*” Oplock break Wait Time can only be configured globally in the `smb.conf` file as shown below.

```
oplock break wait time = 0 (default)
```

`Oplock break contention limit` is an `smb.conf` parameter that limits the response of the Samba server to grant an oplock if the configured number of contending clients reaches the limit specified by the parameter. Samba recommends “*Do not change this parameter unless you have read and understood the Samba oplock code.*” Oplock break Contention Limit can be enable on a per-share basis, or globally for the entire server, in the `smb.conf` file as shown in Example 13.2.

Example 13.2. Configuration with oplock break contention limit

```
[global]
oplock break contention limit = 2 (default)

[share_name]
oplock break contention limit = 2 (default)
```

13.4 MS Windows Opportunistic Locking and Caching Controls

There is a known issue when running applications (like Norton Anti-Virus) on a Windows 2000/ XP workstation computer that can affect any application attempting to access shared database files across a network. This is a result of a default setting configured in the Windows 2000/XP operating system known as *opportunistic locking*. When a workstation attempts to access shared data files located on another Windows 2000/XP computer, the Windows 2000/XP operating system will attempt to increase performance by locking the files and caching information locally. When this occurs, the application is unable to properly function, which results in an “*Access Denied*” error message being displayed during network operations.

All Windows operating systems in the NT family that act as database servers for data files (meaning that data files are stored there and accessed by other Windows PCs) may need to have opportunistic locking disabled in order to minimize the risk of data file corruption. This includes Windows 9x/Me, Windows NT, Windows 200x, and Windows XP.¹

If you are using a Windows NT family workstation in place of a server, you must also disable opportunistic locking (oplocks) on that workstation. For example, if you use a PC with the Windows NT Workstation operating system instead of Windows NT Server, and you have data files located on it that are accessed from other Windows PCs, you may need to disable oplocks on that system.

The major difference is the location in the Windows registry where the values for disabling oplocks are entered. Instead of the LanManServer location, the LanManWorkstation location may be used.

You can verify (change or add, if necessary) this registry value using the Windows Registry Editor. When you change this registry value, you will have to reboot the PC to ensure that the new setting goes into effect.

The location of the client registry entry for opportunistic locking has changed in Windows 2000 from the earlier location in Microsoft Windows NT.

NOTE



Windows 2000 will still respect the EnableOplocks registry value used to disable oplocks in earlier versions of Windows.

You can also deny the granting of opportunistic locks by changing the following registry entries:

```
HKEY_LOCAL_MACHINE\System\  
CurrentControlSet\Services\MRXSmb\Parameters\  
  
OplocksDisabled REG_DWORD 0 or 1  
Default: 0 (not disabled)
```

NOTE



The OplocksDisabled registry value configures Windows clients to either request or not request opportunistic locks on a remote file. To disable oplocks, the value of OplocksDisabled must be set to 1.

¹Microsoft has documented this in Knowledge Base article 300216.

```
HKEY_LOCAL_MACHINE\System\  
CurrentControlSet\Services\LanmanServer\Parameters
```

```
EnableOplocks REG_DWORD 0 or 1  
Default: 1 (Enabled by Default)
```

```
EnableOpLockForceClose REG_DWORD 0 or 1  
Default: 0 (Disabled by Default)
```

NOTE



The EnableOplocks value configures Windows-based servers (including Workstations sharing files) to allow or deny opportunistic locks on local files.

To force closure of open oplocks on close or program exit, EnableOpLockForceClose must be set to 1.

An illustration of how Level2 oplocks work:

- Station 1 opens the file requesting oplock.
- Since no other station has the file open, the server grants station 1 exclusive oplock.
- Station 2 opens the file requesting oplock.
- Since station 1 has not yet written to the file, the server asks station 1 to break to Level2 oplock.
- Station 1 complies by flushing locally buffered lock information to the server.
- Station 1 informs the server that it has Broken to Level2 Oplock (alternately, station 1 could have closed the file).
- The server responds to station 2's open request, granting it Level2 oplock. Other stations can likewise open the file and obtain Level2 oplock.
- Station 2 (or any station that has the file open) sends a write request SMB. The server returns the write response.
- The server asks all stations that have the file open to break to none, meaning no station holds any oplock on the file. Because the workstations can have no cached writes or locks at this point, they need not respond to the break-to-none advisory; all they need do is invalidate locally cached read-ahead data.

13.4.1 Workstation Service Entries

```
\HKEY_LOCAL_MACHINE\System\  
  CurrentControlSet\Services\LanmanWorkstation\Parameters
```

```
UseOpportunisticLocking  REG_DWORD  0 or 1  
Default: 1 (true)
```

This indicates whether the redirector should use opportunistic-locking (oplock) performance enhancement. This parameter should be disabled only to isolate problems.

13.4.2 Server Service Entries

```
\HKEY_LOCAL_MACHINE\System\  
  CurrentControlSet\Services\LanmanServer\Parameters
```

```
EnableOplocks  REG_DWORD  0 or 1  
Default: 1 (true)
```

This specifies whether the server allows clients to use oplocks on files. Oplocks are a significant performance enhancement, but have the potential to cause lost cached data on some networks, particularly wide area networks.

```
MinLinkThroughput  REG_DWORD  0 to infinite bytes per second  
Default: 0
```

This specifies the minimum link throughput allowed by the server before it disables raw and opportunistic locks for this connection.

```
MaxLinkDelay  REG_DWORD  0 to 100,000 seconds  
Default: 60
```

This specifies the maximum time allowed for a link delay. If delays exceed this number, the server disables raw I/O and opportunistic locking for this connection.

```
OplockBreakWait  REG_DWORD  10 to 180 seconds  
Default: 35
```

This specifies the time that the server waits for a client to respond to an oplock break request. Smaller values can allow detection of crashed clients more quickly but can potentially cause loss of cached data.

13.5 Persistent Data Corruption

If you have applied all of the settings discussed in this chapter but data corruption problems and other symptoms persist, here are some additional things to check out.

We have credible reports from developers that faulty network hardware, such as a single faulty network card, can cause symptoms similar to read caching and data corruption. If you see persistent data corruption even after repeated reindexing, you may have to rebuild the data files in question. This involves creating a new data file with the same definition as the file to be rebuilt and transferring the data from the old file to the new one. There are several known methods for doing this that can be found in our Knowledge Base.

13.6 Common Errors

In some sites, locking problems surface as soon as a server is installed; in other sites locking problems may not surface for a long time. Almost without exception, when a locking problem does surface it will cause embarrassment and potential data corruption.

Over the past few years there have been a number of complaints on the Samba mailing lists that have claimed that Samba caused data corruption. Three causes have been identified so far:

- Incorrect configuration of opportunistic locking (incompatible with the application being used. This is a common problem even where MS Windows NT4 or MS Windows 200x-based servers were in use. It is imperative that the software application vendors' instructions for configuration of file locking should be followed. If in doubt, disable oplocks on both the server and the client. Disabling of all forms of file caching on the MS Windows client may be necessary also.
- Defective network cards, cables, or HUBs/Switched. This is generally a more prevalent factor with low cost networking hardware, although occasionally there have also been problems with incompatibilities in more up-market hardware.
- There have been some random reports of Samba log files being written over data files. This has been reported by very few sites (about five in the past three years) and all attempts to reproduce the problem have failed. The Samba Team has been unable to catch this happening and thus has not been able to isolate any particular cause. Considering the millions of systems that use Samba, for the sites that have been affected by this as well as for the Samba Team this is a frustrating and a vexing challenge. If you see this type of thing happening, please create a bug report on Samba Bugzilla² without delay. Make sure that you give as much information as you possibly can help isolate the cause and to allow replication of the problem (an essential step in problem isolation and correction).

13.6.1 locking.tdb Error Messages

“We are seeing lots of errors in the Samba logs, like:

²<https://bugzilla.samba.org>

```
tdb(/usr/local/samba_2.2.7/var/locks/locking.tdb): rec_read bad magic
0x4d6f4b61 at offset=36116
```

What do these mean?

This error indicated a corrupted tdb. Stop all instances of smbd, delete locking.tdb, and restart smbd.

13.6.2 Problems Saving Files in MS Office on Windows XP

This is a bug in Windows XP. More information can be found in Microsoft Knowledge Base article 812937.³

13.6.3 Long Delays Deleting Files Over Network with XP SP1

“It sometimes takes approximately 35 seconds to delete files over the network after XP SP1 has been applied.”

This is a bug in Windows XP. More information can be found in Microsoft Knowledge Base article 811492.⁴

13.7 Additional Reading

You may want to check for an updated version of this white paper on our Web site from time to time. Many of our white papers are updated as information changes. For those papers, the last edited date is always at the top of the paper.

Section of the Microsoft MSDN Library on opportunistic locking:

Opportunistic Locks, Microsoft Developer Network (MSDN), Windows Development > Windows Base Services > Files and I/O > SDK Documentation > File Storage > File Systems > About File Systems > Opportunistic Locks, Microsoft Corporation. http://msdn.microsoft.com/library/en-us/fileio/storage_5yk3.asp

Microsoft Knowledge Base Article Q224992 “*Maintaining Transactional Integrity with OPLOCKS*”, Microsoft Corporation, April 1999, <http://support.microsoft.com/default.aspx?scid=kb;en-us;Q224992>.

Microsoft Knowledge Base Article Q296264 “*Configuring Opportunistic Locking in Windows 2000*”, Microsoft Corporation, April 2001, <http://support.microsoft.com/default.aspx?scid=kb;en-us;Q296264>.

Microsoft Knowledge Base Article Q129202 “*PC Ext: Explanation of Opportunistic Locking on Windows NT*”, Microsoft Corporation, April 1995, <http://support.microsoft.com/default.aspx?scid=kb;en-us;Q129202>.

³<http://support.microsoft.com/?id=812937>

⁴<http://support.microsoft.com/?id=811492>

SECURING SAMBA

14.1 Introduction

This note was attached to the Samba 2.2.8 release notes as it contained an important security fix. The information contained here applies to Samba installations in general.

A new apprentice reported for duty to the chief engineer of a boiler house. He said, “*Here I am, if you will show me the boiler I’ll start working on it.*” Then engineer replied, “*You’re leaning on it!*”

Security concerns are just like that. You need to know a little about the subject to appreciate how obvious most of it really is. The challenge for most of us is to discover that first morsel of knowledge with which we may unlock the secrets of the masters.

14.2 Features and Benefits

There are three levels at which security principals must be observed in order to render a site at least moderately secure. They are the perimeter firewall, the configuration of the host server that is running Samba and Samba itself.

Samba permits a most flexible approach to network security. As far as possible Samba implements the latest protocols to permit more secure MS Windows file and print operations.

Samba may be secured from connections that originate from outside the local network. This may be done using *host-based protection* (using samba’s implementation of a technology known as “*tcpwrappers*,” or it may be done by using *interface-based exclusion* so `smbd` will bind only to specifically permitted interfaces. It is also possible to set specific share or resource-based exclusions, for example on the `[IPC$]` auto-share. The `[IPC$]` share is used for browsing purposes as well as to establish TCP/IP connections.

Another method by which Samba may be secured is by setting Access Control Entries (ACEs) in an Access Control List (ACL) on the shares themselves. This is discussed in Chapter 12, *File, Directory and Share Access Controls*.

14.3 Technical Discussion of Protective Measures and Issues

The key challenge of security is the fact that protective measures suffice at best only to close the door on known exploits and breach techniques. Never assume that because you have followed these few measures that the Samba server is now an impenetrable fortress! Given the history of information systems so far, it is only a matter of time before someone will find yet another vulnerability.

14.3.1 Using Host-Based Protection

In many installations of Samba, the greatest threat comes from outside your immediate network. By default, Samba will accept connections from any host, which means that if you run an insecure version of Samba on a host that is directly connected to the Internet you can be especially vulnerable.

One of the simplest fixes in this case is to use the *hosts allow* and *hosts deny* options in the Samba `smb.conf` configuration file to only allow access to your server from a specific range of hosts. An example might be:

```
hosts allow = 127.0.0.1 192.168.2.0/24 192.168.3.0/24
hosts deny = 0.0.0.0/0
```

The above will only allow SMB connections from `localhost` (your own computer) and from the two private networks 192.168.2 and 192.168.3. All other connections will be refused as soon as the client sends its first packet. The refusal will be marked as not listening or called name error.

14.3.2 User-Based Protection

If you want to restrict access to your server to valid users only, then the following method may be of use. In the `smb.conf` *[global]* section put:

```
valid users = @smbusers, jacko
```

This restricts all server access to either the user *jacko* or to members of the system group *smbusers*.

14.3.3 Using Interface Protection

By default, Samba will accept connections on any network interface that it finds on your system. That means if you have a ISDN line or a PPP connection to the Internet then Samba will accept connections on those links. This may not be what you want.

You can change this behavior using options like this:

```
interfaces = eth* lo
bind interfaces only = yes
```

This tells Samba to only listen for connections on interfaces with a name starting with `eth` such as `eth0`, `eth1` plus on the loopback interface called `lo`. The name you will need to

use depends on what OS you are using. In the above, I used the common name for Ethernet adapters on Linux.

If you use the above and someone tries to make an SMB connection to your host over a PPP interface called `ppp0`, then they will get a TCP connection refused reply. In that case, no Samba code is run at all as the operating system has been told not to pass connections from that interface to any Samba process.

14.3.4 Using a Firewall

Many people use a firewall to deny access to services they do not want exposed outside their network. This can be a good idea, although I recommend using it in conjunction with the above methods so you are protected even if your firewall is not active for some reason.

If you are setting up a firewall, you need to know what TCP and UDP ports to allow and block. Samba uses the following:

- UDP/137 - used by `nmbd`
- UDP/138 - used by `nmbd`
- TCP/139 - used by `smbd`
- TCP/445 - used by `smbd`

The last one is important as many older firewall setups may not be aware of it, given that this port was only added to the protocol in recent years.

14.3.5 Using IPC\$ Share-Based Denials

If the above methods are not suitable, then you could also place a more specific deny on the IPC\$ share that is used in the recently discovered security hole. This allows you to offer access to other shares while denying access to IPC\$ from potentially untrustworthy hosts.

To do this you could use:

```
hosts allow = 192.168.115.0/24 127.0.0.1
hosts deny = 0.0.0.0/0
```

This instructs Samba that IPC\$ connections are not allowed from anywhere except from the two listed network addresses (localhost and the 192.168.115 subnet). Connections to other shares are still allowed. As the IPC\$ share is the only share that is always accessible anonymously, this provides some level of protection against attackers that do not know a valid username/password for your host.

If you use this method, then clients will be given an ‘access denied’ reply when they try to access the IPC\$ share. Those clients will not be able to browse shares, and may also be unable to access some other resources. This is not recommended unless you cannot use one of the other methods listed above for some reason.

14.3.6 NTLMv2 Security

To configure NTLMv2 authentication, the following registry keys are worth knowing about:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa]
"lmcompatibilitylevel"=dword:00000003
```

The value 0x00000003 means send NTLMv2 response only. Clients will use NTLMv2 authentication, use NTLMv2 session security if the server supports it. Domain Controllers accept LM, NTLM and NTLMv2 authentication.

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\MSV1_0]
"NtlmMinClientSec"=dword:00080000
```

The value 0x00080000 means permit only NTLMv2 session security. If either NtlmMinClientSec or NtlmMinServerSec is set to 0x00080000, the connection will fail if NTLMv2 session security is not negotiated.

14.4 Upgrading Samba

Please check regularly on <http://www.samba.org/> for updates and important announcements. Occasionally security releases are made and it is highly recommended to upgrade Samba when a security vulnerability is discovered. Check with your OS vendor for OS specific upgrades.

14.5 Common Errors

If all of Samba and host platform configuration were really as intuitive as one might like them to be, this section would not be necessary. Security issues are often vexing for a support person to resolve, not because of the complexity of the problem, but for the reason that most administrators who post what turns out to be a security problem request are totally convinced that the problem is with Samba.

14.5.1 Smbclient Works on Localhost, but the Network Is Dead

This is a common problem. Red Hat Linux (and others) installs a default firewall. With the default firewall in place, only traffic on the loopback adapter (IP address 127.0.0.1) is allowed through the firewall.

The solution is either to remove the firewall (stop it) or modify the firewall script to allow SMB networking traffic through. See section above in this chapter.

14.5.2 Why Can Users Access Home Directories of Other Users?

“We are unable to keep individual users from mapping to any other user’s home directory once they have supplied a valid password! They only need to enter their own password. I

have not found any method to configure Samba so that users may map only their own home directory.”

“User xyzzy can map his home directory. Once mapped user xyzzy can also map anyone else’s home directory.”

This is not a security flaw, it is by design. Samba allows users to have exactly the same access to the UNIX file system as when they were logged onto the UNIX box, except that it only allows such views onto the file system as are allowed by the defined shares.

If your UNIX home directories are set up so that one user can happily **cd** into another users directory and execute **ls**, the UNIX security solution is to change file permissions on the user’s home directories such that the **cd** and **ls** are denied.

Samba tries very hard not to second guess the UNIX administrators security policies, and trusts the UNIX admin to set the policies and permissions he or she desires.

Samba allows the behavior you require. Simply put the *only user = %S* option in the *[homes]* share definition.

The *only user* works in conjunction with the *users = list*, so to get the behavior you require, add the line :

```
users = %S
```

this is equivalent to adding

```
valid users = %S
```

to the definition of the *[homes]* share, as recommended in the `smb.conf` man page.

INTERDOMAIN TRUST RELATIONSHIPS

Samba-3 supports NT4-style domain trust relationships. This is a feature that many sites will want to use if they migrate to Samba-3 from an NT4-style domain and do not want to adopt Active Directory or an LDAP-based authentication backend. This section explains some background information regarding trust relationships and how to create them. It is now possible for Samba-3 to trust NT4 (and vice versa), as well as to create Samba-to-Samba trusts.

15.1 Features and Benefits

Samba-3 can participate in Samba-to-Samba as well as in Samba-to-MS Windows NT4-style trust relationships. This imparts to Samba similar scalability as with MS Windows NT4.

Given that Samba-3 has the capability to function with a scalable backend authentication database such as LDAP, and given its ability to run in Primary as well as Backup Domain Control modes, the administrator would be well advised to consider alternatives to the use of Interdomain trusts simply because by the very nature of how this works it is fragile. That was, after all, a key reason for the development and adoption of Microsoft Active Directory.

15.2 Trust Relationship Background

MS Windows NT3/4 type security domains employ a non-hierarchical security structure. The limitations of this architecture as it effects the scalability of MS Windows networking in large organizations is well known. Additionally, the flat namespace that results from this design significantly impacts the delegation of administrative responsibilities in large and diverse organizations.

Microsoft developed Active Directory Service (ADS), based on Kerberos and LDAP, as a means of circumventing the limitations of the older technologies. Not every organization is ready or willing to embrace ADS. For small companies the older NT4-style domain security paradigm is quite adequate, there remains an entrenched user base for whom there is no direct desire to go through a disruptive change to adopt ADS.

With MS Windows NT, Microsoft introduced the ability to allow differing security domains to effect a mechanism so users from one domain may be given access rights and privileges in another domain. The language that describes this capability is couched in terms of *Trusts*. Specifically, one domain will *trust* the users from another domain. The domain from which users are available to another security domain is said to be a trusted domain. The domain in which those users have assigned rights and privileges is the trusting domain. With NT3.x/4.0 all trust relationships are always in one direction only, thus if users in both domains are to have privileges and rights in each others' domain, then it is necessary to establish two relationships, one in each direction.

In an NT4-style MS security domain, all trusts are non-transitive. This means that if there are three domains (let's call them RED, WHITE and BLUE) where RED and WHITE have a trust relationship, and WHITE and BLUE have a trust relationship, then it holds that there is no implied trust between the RED and BLUE domains. Relationships are explicit and not transitive.

New to MS Windows 2000 ADS security contexts is the fact that trust relationships are two-way by default. Also, all inter-ADS domain trusts are transitive. In the case of the RED, WHITE and BLUE domains above, with Windows 2000 and ADS the RED and BLUE domains can trust each other. This is an inherent feature of ADS domains. Samba-3 implements MS Windows NT4-style Interdomain trusts and interoperates with MS Windows 200x ADS security domains in similar manner to MS Windows NT4-style domains.

15.3 Native MS Windows NT4 Trusts Configuration

There are two steps to creating an interdomain trust relationship. To effect a two-way trust relationship, it is necessary for each domain administrator to create a trust account for the other domain to use in verifying security credentials.

15.3.1 Creating an NT4 Domain Trust

For MS Windows NT4, all domain trust relationships are configured using the Domain User Manager. This is done from the Domain User Manager Policies entry on the menu bar. From the **Policy** menu, select **Trust Relationships**. Next to the lower box labeled **Permitted to Trust this Domain** are two buttons, **Add** and **Remove**. The **Add** button will open a panel in which to enter the name of the remote domain that will be able to assign access rights to users in your domain. You will also need to enter a password for this trust relationship, which the trusting domain will use when authenticating users from the trusted domain. The password needs to be typed twice (for standard confirmation).

15.3.2 Completing an NT4 Domain Trust

A trust relationship will work only when the other (trusting) domain makes the appropriate connections with the trusted domain. To consummate the trust relationship, the administrator will launch the Domain User Manager from the menu select **Policies**, then select **Trust Relationships**, click on the **Add** button next to the box that is labeled **Trusted Domains**.

A panel will open in which must be entered the name of the remote domain as well as the password assigned to that trust.

15.3.3 Inter-Domain Trust Facilities

A two-way trust relationship is created when two one-way trusts are created, one in each direction. Where a one-way trust has been established between two MS Windows NT4 domains (let's call them DomA and DomB), the following facilities are created:

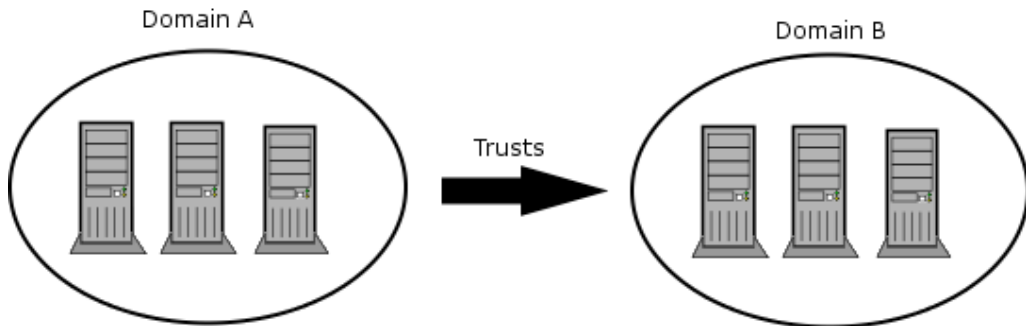


Figure 15.1. Trusts overview.

- DomA (completes the trust connection) *Trusts* DomB.
- DomA is the *Trusting* domain.
- DomB is the *Trusted* domain (originates the trust account).
- Users in DomB can access resources in DomA.
- Users in DomA cannot access resources in DomB.
- Global groups from DomB can be used in DomA.
- Global groups from DomA cannot be used in DomB.
- DomB does appear in the logon dialog box on client workstations in DomA.
- DomA does not appear in the logon dialog box on client workstations in DomB.
- Users/Groups in a trusting domain cannot be granted rights, permissions or access to a trusted domain.
- The trusting domain can access and use accounts (Users/Global Groups) in the trusted domain.
- Administrators of the trusted domain can be granted administrative rights in the trusting domain.
- Users in a trusted domain can be given rights and privileges in the trusting domain.
- Trusted domain Global Groups can be given rights and permissions in the trusting domain.

- Global Groups from the trusted domain can be made members in Local Groups on MS Windows Domain Member machines.

15.4 Configuring Samba NT-Style Domain Trusts

This description is meant to be a fairly short introduction about how to set up a Samba server so that it can participate in interdomain trust relationships. Trust relationship support in Samba is at an early stage, so do not be surprised if something does not function as it should.

Each of the procedures described below assumes the peer domain in the trust relationship is controlled by a Windows NT4 server. However, the remote end could just as well be another Samba-3 domain. It can be clearly seen, after reading this document, that combining Samba-specific parts of what's written below leads to trust between domains in a purely Samba environment.

15.4.1 Samba as the Trusted Domain

In order to set the Samba PDC to be the trusted party of the relationship, you first need to create a special account for the domain that will be the trusting party. To do that, you can use the **smbpasswd** utility. Creating the trusted domain account is similar to creating a trusted machine account. Suppose, your domain is called SAMBA, and the remote domain is called RUMBA. The first step will be to issue this command from your favorite shell:

```
root# smbpasswd -a -i rumba
New SMB password: XXXXXXXX
Retype SMB password: XXXXXXXX
Added user rumba$
```

where **-a** means to add a new account into the `passwd` database and **-i** means: “*create this account with the InterDomain trust flag*”.

The account name will be “*rumba\$*” (the name of the remote domain). If this fails, you should check that the trust account has been added to the system password database (`/etc/passwd`). If it has not been added, you can add it manually and then repeat the step above.

After issuing this command, you will be asked to enter the password for the account. You can use any password you want, but be aware that Windows NT will not change this password until seven days following account creation. After the command returns successfully, you can look at the entry for the new account (in the standard way as appropriate for your configuration) and see that account's name is really RUMBA\$ and it has the “*T*” flag set in the flags field. Now you are ready to confirm the trust by establishing it from Windows NT Server.

Open User Manager for Domains and from the **Policies** menu, select **Trust Relationships....** Beside the **Trusted domains** list box click the **Add...** button. You will be prompted for the trusted domain name and the relationship password. Type in SAMBA, as this is the name

of the remote domain and the password used at the time of account creation. Click on **OK** and, if everything went without incident, you will see the **Trusted domain relationship successfully established** message.

15.4.2 Samba as the Trusting Domain

This time activities are somewhat reversed. Again, we'll assume that your domain controlled by the Samba PDC is called SAMBA and the NT-controlled domain is called RUMBA.

The very first step is to add an account for the SAMBA domain on RUMBA's PDC.

Launch the Domain User Manager, then from the menu select **Policies, Trust Relationships**. Now, next to the **Trusted Domains** box press the **Add** button and type in the name of the trusted domain (SAMBA) and the password to use in securing the relationship.

The password can be arbitrarily chosen. It is easy to change the password from the Samba server whenever you want. After confirming the password your account is ready for use. Now it's Samba's turn.

Using your favorite shell while being logged in as root, issue this command:

```
root#net rpc trustdom establish rumba
```

You will be prompted for the password you just typed on your Windows NT4 Server box. An error message 'NT_STATUS_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT' that may be reported periodically is of no concern and may safely be ignored. It means the password you gave is correct and the NT4 Server says the account is ready for interdomain connection and not for ordinary connection. After that, be patient; it can take a while (especially in large networks), but eventually you should see the **Success** message. Congratulations! Your trust relationship has just been established.

NOTE



You have to run this command as root because you must have write access to the `secrets.tdb` file.

15.5 NT4-Style Domain Trusts with Windows 2000

Although Domain User Manager is not present in Windows 2000, it is also possible to establish an NT4-style trust relationship with a Windows 2000 domain controller running in mixed mode as the trusting server. It should also be possible for Samba to trust a Windows 2000 server, however, more testing is still needed in this area.

After creating the interdomain trust account on the Samba server as described above, open Active Directory Domains and Trusts on the AD controller of the domain whose resources you wish Samba users to have access to. Remember that since NT4-style trusts are not transitive, if you want your users to have access to multiple mixed-mode domains in your AD

forest, you will need to repeat this process for each of those domains. With Active Directory Domains and Trusts open, right-click on the name of the Active Directory domain that will trust our Samba domain and choose **Properties**, then click on the **Trusts** tab. In the upper part of the panel, you will see a list box labeled **Domains trusted by this domain:**, and an **Add...** button next to it. Press this button and just as with NT4, you will be prompted for the trusted domain name and the relationship password. Press OK and after a moment, Active Directory will respond with **The trusted domain has been added and the trust has been verified**. Your Samba users can now be granted access to resources in the AD domain.

15.6 Common Errors

Interdomain trust relationships should not be attempted on networks that are unstable or that suffer regular outages. Network stability and integrity are key concerns with distributed trusted domains.

15.6.1 Browsing of Trusted Domain Fails

Browsing from a machine in a trusted Windows 200x Domain to a Windows 200x member of a trusting samba domain, I get the following error:

The system detected a possible attempt to compromise security. Please ensure that you can contact the server that authenticated you.

The event logs on the box I'm trying to connect to have entries regarding group policy not being applied because it is a member of a downlevel domain.

Answer: If there is a computer account in the Windows 200x Domain for the machine in question, and it is disabled, this problem rears can occur. If there is no computer account (removed or never existed), or if that account is still intact (i.e.: you just joined it to another domain) everything seems to be fine. By default, when you unjoin a domain (the Windows 200x Domain), the computer tries to automatically disable the computer account in the domain. If you are running as an account which has privileges to do this when you unjoin the machine, it is done, otherwise it is not done.

HOSTING A MICROSOFT DISTRIBUTED FILE SYSTEM TREE

16.1 Features and Benefits

The Distributed File System (DFS) provides a means of separating the logical view of files and directories that users see from the actual physical locations of these resources on the network. It allows for higher availability, smoother storage expansion, load balancing, and so on.

For information about DFS, refer to the Microsoft documentation¹. This document explains how to host a DFS tree on a UNIX machine (for DFS-aware clients to browse) using Samba.

To enable SMB-based DFS for Samba, configure it with the `--with-msdfs` option. Once built, a Samba server can be made a DFS server by setting the global Boolean `host msdfs` parameter in the `smb.conf` file. You designate a share as a DFS root using the Share Level Boolean `msdfs root` parameter. A DFS root directory on Samba hosts DFS links in the form of symbolic links that point to other servers. For example, a symbolic link `junction->msdfs:storage1\share1` in the share directory acts as the DFS junction. When DFS-aware clients attempt to access the junction link, they are redirected to the storage location (in this case, `\\storage1\share1`).

DFS trees on Samba work with all DFS-aware clients ranging from Windows 95 to 200x. Example 16.1 shows how to setup a DFS tree on a Samba server. In the `/export/dfsroot` directory, you set up your DFS links to other servers on the network.

```
root# cd /export/dfsroot
root# chown root /export/dfsroot
root# chmod 755 /export/dfsroot
root# ln -s msdfs:storageA\shareA linka
root# ln -s msdfs:serverB\share,serverC\share linkb
```

¹<http://www.microsoft.com/NTServer/nts/downloads/winfeatures/NTSDistrFile/AdminGuide.asp>

Example 16.1. smb.conf with DFS configured

```
[global]
netbios name = GANDALF
host msdfs = yes

[dfs]
path = /export/dfsroot
msdfs root = yes
```

You should set up the permissions and ownership of the directory acting as the DFS root so that only designated users can create, delete or modify the msdfs links. Also note that symlink names should be all lowercase. This limitation exists to have Samba avoid trying all the case combinations to get at the link name. Finally, set up the symbolic links to point to the network shares you want and start Samba.

Users on DFS-aware clients can now browse the DFS tree on the Samba server at `\\samba\dfs`. Accessing links `linka` or `linkb` (which appear as directories to the client) takes users directly to the appropriate shares on the network.

16.2 Common Errors

- Windows clients need to be rebooted if a previously mounted non-DFS share is made a DFS root or vice versa. A better way is to introduce a new share and make it the DFS root.
- Currently, there's a restriction that msdfs symlink names should all be lowercase.
- For security purposes, the directory acting as the root of the DFS tree should have ownership and permissions set so only designated users can modify the symbolic links in the directory.

16.2.1 MSDFS UNIX Path Is Case-Critical

A network administrator sent advice to the Samba mailing list after a long sessions trying to determine why DFS was not working. His advice is worth noting.

“I spent some time trying to figure out why my particular dfs root wasn't working. I noted in the documentation that the symlink should be in all lowercase. It should be amended that the entire path to the symlink should all be in lowercase as well.”

For example, I had a share defined as such:

```
[pub]
path = /export/home/Shares/public_share
msdfs root = yes
```


and I could not make my Windows 9x/Me (with the dfs client installed) follow this symlink:

```
damage1 -> msdfs:damage\test-share
```

Running a debug level of 10 reveals:

```
[2003/08/20 11:40:33, 5] msdfs/msdfs.c:is_msdfs_link(176)
  is_msdfs_link: /export/home/shares/public_share/* does not exist.
```

Curious. So I changed the directory name from .../Shares/... to .../shares/... (along with my service definition) and it worked!

CLASSICAL PRINTING SUPPORT

17.1 Features and Benefits

Printing is often a mission-critical service for the users. Samba can provide this service reliably and seamlessly for a client network consisting of Windows workstations.

A Samba print service may be run on a Stand-alone or Domain Member server, side by side with file serving functions, or on a dedicated print server. It can be made as tight or as loosely secured as needs dictate. Configurations may be simple or complex. Available authentication schemes are essentially the same as described for file services in previous chapters. Overall, Samba's printing support is now able to replace an NT or Windows 2000 print server full-square, with additional benefits in many cases. Clients may download and install drivers and printers through their familiar "*Point'n'Print*" mechanism. Printer installations executed by "*Logon Scripts*" are no problem. Administrators can upload and manage drivers to be used by clients through the familiar "*Add Printer Wizard*". As an additional benefit, driver and printer management may be run from the command line or through scripts, making it more efficient in case of large numbers of printers. If a central accounting of print jobs (tracking every single page and supplying the raw data for all sorts of statistical reports) is required, this function is best supported by the newer Common UNIX Printing System (CUPS) as the print subsystem underneath the Samba hood.

This chapter deals with the foundations of Samba printing as they are implemented by the more traditional UNIX (BSD- and System V-style) printing systems. Many things covered in this chapter apply also to CUPS. If you use CUPS, you may be tempted to jump to the next chapter but you will certainly miss a few things if you do. It is recommended that you read this chapter as well as Chapter 18, *CUPS Printing Support*.

NOTE



Most of the following examples have been verified on Windows XP Professional clients. Where this document describes the responses to commands given, bear in mind that Windows 200x/XP clients are quite similar, but may differ in minor details. Windows NT is somewhat different again.

17.2 Technical Introduction

Samba's printing support always relies on the installed print subsystem of the UNIX OS it runs on. Samba is a “*middleman*.” It takes print files from Windows (or other SMB) clients and passes them to the real printing system for further processing, therefore, it needs to communicate with both sides: the Windows print clients and the UNIX printing system. Hence, we must differentiate between the various client OS types, each of which behave differently, as well as the various UNIX print subsystems, which themselves have different features and are accessed differently.

This deals with the traditional way of UNIX printing. The next chapter covers in great detail the more modern *Common UNIX Printing System* (CUPS).

IMPORTANT



CUPS users, be warned: do not just jump on to the next chapter. You might miss important information only found here!

It is apparent from postings on the Samba mailing list that print configuration is one of the most problematic aspects of Samba administration today. Many new Samba administrators have the impression that Samba performs some sort of print processing. Rest assured, Samba does not perform any type of print processing. It does not do any form of print filtering.

Samba obtains from its clients a data stream (print job) that it spools to a local spool area. When the entire print job has been received, Samba invokes a local UNIX/Linux print command and passes the spooled file to it. It is up to the local system printing subsystems to correctly process the print job and to submit it to the printer.

17.2.1 Client to Samba Print Job Processing

Successful printing from a Windows client via a Samba print server to a UNIX printer involves six (potentially seven) stages:

1. Windows opens a connection to the printer share.
2. Samba must authenticate the user.

3. Windows sends a copy of the print file over the network into Samba's spooling area.
4. Windows closes the connection.
5. Samba invokes the print command to hand the file over to the UNIX print subsystem's spooling area.
6. The UNIX print subsystem processes the print job.
7. The print file may need to be explicitly deleted from the Samba spooling area. This item depends on your print spooler configuration settings.

17.2.2 Printing Related Configuration Parameters

There are a number of configuration parameters to control Samba's printing behavior. Please refer to the man page for `smb.conf` for an overview of these. As with other parameters, there are Global Level (tagged with a *G* in the listings) and Service Level (*S*) parameters.

Global Parameters — These *may not* go into individual share definitions. If they go in by error, the `testparm` utility can discover this (if you run it) and tell you so.

Service Level Parameters — These may be specified in the `[global]` section of `smb.conf`. In this case they define the default behavior of all individual or service level shares (provided they do not have a different setting defined for the same parameter, thus overriding the global default).

17.3 Simple Print Configuration

Example 17.1 shows a simple printing configuration. If you compare this with your own, you may find additional parameters that have been pre-configured by your OS vendor. Below is a discussion and explanation of the parameters. This example does not use many parameters. However, in many environments these are enough to provide a valid `smb.conf` file that enables all clients to print.

Example 17.1. Simple configuration with BSD printing

```
[global]
printing = bsd
load printers = yes

[printers]
path = /var/spool/samba
printable = yes
public = yes
writable = no
```

This is only an example configuration. Samba assigns default values to all configuration parameters. The defaults are conservative and sensible. When a parameter is specified in the `smb.conf` file, this overwrites the default value. The `testparm` utility when run as root is capable of reporting all settings, both default as well as `smb.conf` file settings. `Testparm` gives warnings for all misconfigured settings. The complete output is easily 340 lines and more, so you may want to pipe it through a pager program.

The syntax for the configuration file is easy to grasp. You should know that is not very picky about its syntax. As has been explained elsewhere in this document, Samba tolerates some spelling errors (such as *browsable* instead of *browseable*), and spelling is case-insensitive. It is permissible to use *Yes/No* or *True/False* for Boolean settings. Lists of names may be separated by commas, spaces or tabs.

17.3.1 Verifying Configuration with testparm

To see all (or at least most) printing-related settings in Samba, including the implicitly used ones, try the command outlined below. This command greps for all occurrences of `lp`, `print`, `spool`, `driver`, `ports` and `[` in testparms output. This provides a convenient overview of the running `smbd` print configuration. This command does not show individually created printer shares or the spooling paths they may use. Here is the output of my Samba setup, with settings shown in Example 17.1:

```
root# testparm -s -v | egrep "(lp|print|spool|driver|ports|\[)"
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"

[global]
    smb ports = 445 139
    lpq cache time = 10
    total print jobs = 0
    load printers = Yes
    printcap name = /etc/printcap
    disable spoolss = No
    enumports command =
    addprinter command =
    deleteprinter command =
    show add printer wizard = Yes
    os2 driver map =
    printer admin =
    min print space = 0
    max print jobs = 1000
    printable = No
    printing = bsd
    print command = lpr -r -P'%p' %s
    lpq command = lpq -P'%p'
    lprm command = lprm -P'%p' %j
```

```
lppause command =
lpresume command =
printer name =
use client driver = No
```

```
[homes]
```

```
[printers]
```

```
path = /var/spool/samba
printable = Yes
```

You can easily verify which settings were implicitly added by Samba's default behavior. *Remember: it may be important in your future dealings with Samba.*

NOTE



testparm in Samba-3 behaves differently from that in 2.2.x: used without the “-v” switch it only shows you the settings actually written into! To see the complete configuration used, add the “-v” parameter to testparm.

17.3.2 Rapid Configuration Validation

Should you need to troubleshoot at any stage, please always come back to this point first and verify if **testparm** shows the parameters you expect. To give you a warning from personal experience, try to just comment out the *load printers* parameter. If your 2.2.x system behaves like mine, you'll see this:

```
root# grep "load printers" /etc/samba/smb.conf
# load printers = Yes
# This setting is commented out!!
```

```
root# testparm -v /etc/samba/smb.conf | egrep "(load printers)"
load printers = Yes
```

I assumed that commenting out of this setting should prevent Samba from publishing my printers, but it still did. It took some time to figure out the reason. But I am no longer fooled ... at least not by this.

```
root# grep -A1 "load printers" /etc/samba/smb.conf
load printers = No
# The above setting is what I want!
# load printers = Yes
```

```
# This setting is commented out!
```

```
root# testparm -s -v smb.conf.simpleprinting | egrep "(load printers)"
load printers = No
```

Only when the parameter is explicitly set to *load printers* = No would Samba conform with my intentions. So, my strong advice is:

- Never rely on commented out parameters.
- Always set parameters explicitly as you intend them to behave.
- Use **testparm** to uncover hidden settings that might not reflect your intentions.

The following is the most minimal configuration file:

```
root# cat /etc/samba/smb.conf-minimal
[printers]
```

This example should show that you can use testparm to test any Samba configuration file. Actually, we encourage you *not* to change your working system (unless you know exactly what you are doing). Don't rely on the assumption that changes will only take effect after you re-start smbd! This is not the case. Samba re-reads it every 60 seconds and on each new client connection. You might have to face changes for your production clients that you didn't intend to apply. You will now note a few more interesting things; **testparm** is useful to identify what the Samba print configuration would be if you used this minimalistic configuration. Here is what you can expect to find:

```
root# testparm -v smb.conf-minimal | egrep "(print|lpq|spool|driver|ports|[])"
Processing section "[printers]"
WARNING: [printers] service MUST be printable!
No path in service printers - using /tmp

lpq cache time = 10
total print jobs = 0
load printers = Yes
printcap name = /etc/printcap
disable spoolss = No
enumports command =
addprinter command =
deleteprinter command =
show add printer wizard = Yes
os2 driver map =
printer admin =
min print space = 0
max print jobs = 1000
printable = No
```



```
printing = bsd
print command = lpr -r -P%p %s
lpq command = lpq -P%p
printer name =
use client driver = No
```

```
[printers]
printable = Yes
```

testparm issued two warnings:

- We did not specify the *[printers]* section as printable.
- We did not tell Samba which spool directory to use.

However, this was not fatal and Samba will default to values that will work. Please, do not rely on this and do not use this example. This was included to encourage you to be careful to design and specify your setup to do precisely what you require. The outcome on your system may vary for some parameters given, since Samba may have been built with different compile-time options. *Warning:* do not put a comment sign *at the end* of a valid line. It will cause the parameter to be ignored (just as if you had put the comment sign at the front). At first I regarded this as a bug in my Samba versions. But the man page clearly says: “*Internal whitespace in a parameter value is retained verbatim.*” This means that a line consisting of, for example:

```
printing = lprng
```

will regard the whole of the string after the “=” sign as the value you want to define. This is an invalid value that will be ignored and a default value will be used in its place.

17.4 Extended Printing Configuration

In Example 17.2 we show a more verbose example configuration for print-related settings in a BSD-style printing environment. What follows is a discussion and explanation of the various parameters. We chose to use BSD-style printing here because it is still the most commonly used system on legacy UNIX/Linux installations. New installations predominantly use CUPS, which is discussed in a separate chapter. Example 17.2 explicitly names many parameters that do not need to be specified because they are set by default. You could use a much leaner `smb.conf` file. Alternately, you can use **testparm** or **SWAT** to optimize the `smb.conf` file to remove all parameters that are set at default.

This is an example configuration. You may not find all the settings that are in the configuration file that was provided by the OS vendor. Samba configuration parameters, if not explicitly set default to a sensible value. To see all settings, as **root** use the **testparm** utility. **testparm** gives warnings for misconfigured settings.

Example 17.2. Extended BSD Printing Configuration

```
[global]
printing = bsd
load printers = yes
show add printer wizard = yes
printcap name = /etc/printcap
printer admin = @ntadmin, root
total print jobs = 100
lpq cache time = 20
use client driver = no

[printers]
comment = All Printers
printable = yes
path = /var/spool/samba
browseable = no
guest ok = yes
public = yes
read only = yes
writable = no

[my_printer_name]
comment = Printer with Restricted Access
path = /var/spool/samba_my_printer
printer admin = kurt
browseable = yes
printable = yes
writeable = no
hosts allow = 0.0.0.0
hosts deny = turbo.xp, 10.160.50.23, 10.160.51.60
guest ok = no
```

17.4.1 Detailed Explanation Settings

The following is a discussion of the settings from above shown example.

17.4.1.1 The [global] Section

The *[global]* section is one of four special sections (along with *[homes]*, *[printers]* and *[print\$]*...). The *[global]* contains all parameters which apply to the server as a whole. It is the place for parameters that have only a global meaning. It may also contain service level parameters that then define default settings for all other sections and shares. This way you can simplify the configuration and avoid setting the same value repeatedly. (Within each individual section or share you may, however, override these globally set share settings and specify other values).

printing = **bsd** — Causes Samba to use default print commands applicable for the BSD (also known as RFC 1179 style or LPR/LPD) printing system. In general, the *printing* parameter informs Samba about the print subsystem it should expect. Samba supports CUPS, LPD, LPRNG, SYSV, HPUX, AIX, QNX, and PLP. Each of these systems defaults to a different *print command* (and other queue control commands).

CAUTION

The *printing* parameter is normally a service level parameter. Since it is included here in the *[global]* section, it will take effect for all printer shares that are not defined differently. Samba-3 no longer supports the SOFTQ printing system.

load printers = **yes** — Tells Samba to create automatically all available printer shares. Available printer shares are discovered by scanning the printcap file. All created printer shares are also loaded for browsing. If you use this parameter, you do not need to specify separate shares for each printer. Each automatically created printer share will clone the configuration options found in the *[printers]* section. (The *load printers* = **no** setting will allow you to specify each UNIX printer you want to share separately, leaving out some you do not want to be publicly visible and available).

show add printer wizard = **yes** — Setting is normally enabled by default (even if the parameter is not specified in *smb.conf*). It causes the **Add Printer Wizard** icon to appear in the **Printers** folder of the Samba host's share listing (as shown in **Network Neighborhood** or by the **net view** command). To disable it, you need to explicitly set it to **no** (commenting it out will not suffice). The *Add Printer Wizard* lets you upload printer drivers to the *[print\$]* share and associate it with a printer (if the respective queue exists before the action), or exchange a printer's driver against any other previously uploaded driver.

total print jobs = **100** — Sets the upper limit to 100 print jobs being active on the Samba server at any one time. Should a client submit a job that exceeds this number, a “*no more space available on server*” type of error message will be returned by Samba to the client. A setting of zero (the default) means there is *no* limit at all.

printcap name = **/etc/printcap** — Tells Samba where to look for a list of available printer names. Where CUPS is used, make sure that a printcap file is written. This is controlled by the **Printcap** directive in the *cupsd.conf* file.

printer admin = **@ntadmin** — Members of the *ntadmin* group should be able to add drivers and set printer properties (*ntadmin* is only an example name, it needs to be a valid UNIX group name); root is implicitly always a *printer admin*. The @ sign

precedes group names in the `/etc/group`. A printer admin can do anything to printers via the remote administration interfaces offered by MS-RPC (see below). In larger installations, the *printer admin* parameter is normally a per-share parameter. This permits different groups to administer each printer share.

lpq cache time = 20 — Controls the cache time for the results of the `lpq` command. It prevents the `lpq` command being called too often and reduces the load on a heavily used print server.

use client driver = no — If set to *yes*, only takes effect for Windows NT/200x/XP clients (and not for Win 95/98/ME). Its default value is *No* (or *False*). It must *not* be enabled on print shares (with a *yes* or *true* setting) that have valid drivers installed on the Samba server. For more detailed explanations see the `smb.conf` man page.

17.4.1.2 The [printers] Section

This is the second special section. If a section with this name appears in the `smb.conf`, users are able to connect to any printer specified in the Samba host's `printcap` file, because Samba on startup then creates a printer share for every printername it finds in the `printcap` file. You could regard this section as a general convenience shortcut to share all printers with minimal configuration. It is also a container for settings that should apply as default to all printers. (For more details see the `smb.conf` man page.) Settings inside this container must be Share Level parameters.

comment = All printers — The *comment* is shown next to the share if a client queries the server, either via **Network Neighborhood** or with the **net view** command to list available shares.

printable = yes — The `[printers]` service *must* be declared as printable. If you specify otherwise, `smbd` will refuse to load at startup. This parameter allows connected clients to open, write to and submit spool files into the directory specified with the *path* parameter for this service. It is used by Samba to differentiate printer shares from file shares.

path = /var/spool/samba — Must point to a directory used by Samba to spool incoming print files. *It must not be the same as the spool directory specified in the configuration of your UNIX print subsystem!* The path typically points to a directory that is world writeable, with the “*sticky*” bit set to it.

browseable = no — Is always set to *no* if *printable = yes*. It makes the `[printer]` share itself invisible in the list of available shares in a **net view** command or in the Explorer browse list. (You will of course see the individual printers).

guest ok = yes — If this parameter is set to **yes**, no password is required to connect to the printer's service. Access will be granted with the privileges of the *guest account*. On many systems the guest account will map to a user named "*nobody*". This user will usually be found in the UNIX `passwd` file with an empty password, but with no valid UNIX login. (On some systems the guest account might not have the privilege to be able to print. Test this by logging in as your guest user using **su - guest** and run a system print command like:

```
lpr -P printername /etc/motd
```

public = yes — Is a synonym for *guest ok = yes*. Since we have *guest ok = yes*, it really does not need to be here. (This leads to the interesting question: "*What if I by accident have two contradictory settings for the same share?*" The answer is the last one encountered by Samba wins. Testparm does not complain about different settings of the same parameter for the same share. You can test this by setting up multiple lines for the *guest account* parameter with different usernames, and then run testparm to see which one is actually used by Samba.)

read only = yes — Normally (for other types of shares) prevents users from creating or modifying files in the service's directory. However, in a "*printable*" service, it is *always* allowed to write to the directory (if user privileges allow the connection), but only via print spooling operations. Normal write operations are not permitted.

writable = no — Is a synonym for *read only = yes*.

17.4.1.3 Any [*my_printer_name*] Section

If a section appears in the `smb.conf` file, which when given the parameter *printable = yes* causes Samba to configure it as a printer share. Windows 9x/Me clients may have problems with connecting or loading printer drivers if the share name has more than eight characters. Do not name a printer share with a name that may conflict with an existing user or file share name. On Client connection requests, Samba always tries to find file shares with that name first. If it finds one, it will connect to this and will not connect to a printer with the same name!

comment = Printer with Restricted Access — The comment says it all.

path = /var/spool/samba.my_printer — Sets the spooling area for this printer to a directory other than the default. It is not necessary to set it differently, but the option is available.

printer admin = kurt — The printer admin definition is different for this explicitly defined printer share from the general [*printers*] share. It is not a requirement; we did it to show that it is possible.

browseable = **yes** — This makes the printer browseable so the clients may conveniently find it when browsing the **Network Neighborhood**.

printable = **yes** — See Section 17.4.1.2.

writable = **no** — See Section 17.4.1.2.

hosts allow = **10.160.50.,10.160.51.** — Here we exercise a certain degree of access control by using the *hosts allow* and *hosts deny* parameters. This is not by any means a safe bet. It is not a way to secure your printers. This line accepts all clients from a certain subnet in a first evaluation of access control.

hosts deny = **turbo_xp,10.160.50.23,10.160.51.60** — All listed hosts are not allowed here (even if they belong to the allowed subnets). As you can see, you could name IP addresses as well as NetBIOS hostnames here.

guest ok = **no** — This printer is not open for the guest account.

17.4.1.4 Print Commands

In each section defining a printer (or in the *[printers]* section), a *print command* parameter may be defined. It sets a command to process the files that have been placed into the Samba print spool directory for that printer. (That spool directory was, if you remember, set up with the *path* parameter). Typically, this command will submit the spool file to the Samba host's print subsystem, using the suitable system print command. But there is no requirement that this needs to be the case. For debugging or some other reason, you may want to do something completely different than print the file. An example is a command that just copies the print file to a temporary location for further investigation when you need to debug printing. If you craft your own print commands (or even develop print command shell scripts), make sure you pay attention to the need to remove the files from the Samba spool directory. Otherwise, your hard disk may soon suffer from shortage of free space.

17.4.1.5 Default UNIX System Printing Commands

You learned earlier on that Samba, in most cases, uses its built-in settings for many parameters if it cannot find an explicitly stated one in its configuration file. The same is true for the *print command*. The default print command varies depending on the *printing* parameter setting. In the commands listed below, you will notice some parameters of the form %X where X is *p*, *s*, *J*, and so on. These letters stand for printer name, spoolfile and job ID, respectively. They are explained in more detail further below. Table 17.1 presents an overview of key printing options but excludes the special case of CUPS that is discussed in Chapter 18, *CUPS Printing Support*.

Table 17.1. Default Printing Settings

Setting	Default Printing Commands
<i>printing</i> = bsd aix lprng plp	print command is lpr -r -P%p %s
<i>printing</i> = sysv hpux	print command is lp -c -P%p %s; rm %s
<i>printing</i> = qnx	print command is lp -r -P%p -s %s
<i>printing</i> = bsd aix lprng plp	lpq command is lpq -P%p
<i>printing</i> = sysv hpux	lpq command is lpstat -o%p
<i>printing</i> = qnx	lpq command is lpq -P%p
<i>printing</i> = bsd aix lprng plp	lprm command is lprm -P%p %j
<i>printing</i> = sysv hpux	lprm command is cancel %p-%j
<i>printing</i> = qnx	lprm command is cancel %p-%j
<i>printing</i> = bsd aix lprng plp	lppause command is lp -i %p-%j -H hold
<i>printing</i> = sysv hpux	lppause command (...is empty)
<i>printing</i> = qnx	lppause command (...is empty)
<i>printing</i> = bsd aix lprng plp	lpresume command is lp -i %p-%j -H resume
<i>printing</i> = sysv hpux	lpresume command (...is empty)
<i>printing</i> = qnx	lpresume command (...is empty)

We excluded the special case of CUPS here, because it is discussed in the next chapter. For *printing* = *CUPS*, if Samba is compiled against libcups, it uses the CUPS API to submit jobs. (It is a good idea also to set *printcap* = cups in case your *cupsd.conf* is set to write its autogenerated *printcap* file to an unusual place). Otherwise, Samba maps to the System V printing commands with the *-oraw* option for printing, i.e., it uses **lp -c -d%p -oraw; rm %s**. With *printing* = *cups*, and if Samba is compiled against libcups, any manually set print command will be ignored!

17.4.1.6 Custom Print Commands

After a print job has finished spooling to a service, the *print command* will be used by Samba via a *system()* call to process the spool file. Usually the command specified will submit the spool file to the host's printing subsystem. But there is no requirement at all that this must be the case. The print subsystem may not remove the spool file on its own. So whatever command you specify, you should ensure that the spool file is deleted after it has been processed.

There is no difficulty with using your own customized print commands with the traditional printing systems. However, if you do not wish to roll your own, you should be well informed about the default built-in commands that Samba uses for each printing subsystem (see Table 17.1). In all the commands listed in the last paragraphs, you see parameters of the form *%X*. These are *macros*, or shortcuts, used as placeholders for the names of real objects. At the time of running a command with such a placeholder, Samba will insert the appropriate value automatically. Print commands can handle all Samba macro substitutions. In regard to printing, the following ones do have special relevance:

- *%s*, *%f* — the path to the spool file name.
- *%p* — the appropriate printer name.

- *%J* — the job name as transmitted by the client.
- *%c* — the number of printed pages of the spooled job (if known).
- *%z* — the size of the spooled print job (in bytes).

The print command must contain at least one occurrence of *%s* or the *%f*. The *%p* is optional. If no printer name is supplied, the *%p* will be silently removed from the print command. In this case, the job is sent to the default printer.

If specified in the *[global]* section, the print command given will be used for any printable service that does not have its own print command specified. If there is neither a specified print command for a printable service nor a global print command, spool files will be created but not processed! Most importantly, print files will not be removed, so they will consume disk space.

Printing may fail on some UNIX systems when using the “*nobody*” account. If this happens, create an alternative guest account and give it the privilege to print. Set up this guest account in the *[global]* section with the *guest account* parameter.

You can form quite complex print commands. You need to realize that print commands are just passed to a UNIX shell. The shell is able to expand the included environment variables as usual. (The syntax to include a UNIX environment variable *\$variable* in the Samba print command is *`\${variable}*.) To give you a working *print command* example, the following will log a print job to */tmp/print.log*, print the file, then remove it. The semicolon (“;” is the usual separator for commands in shell scripts:

```
print command = echo Printing %s >> /tmp/print.log; lpr -P %p %s; rm %s
```

You may have to vary your own command considerably from this example depending on how you normally print files on your system. The default for the *print command* parameter varies depending on the setting of the *printing* parameter. Another example is:

```
print command = /usr/local/samba/bin/myprintscript %p %s
```

17.5 Printing Developments Since Samba-2.2

Prior to Samba-2.2.x, print server support for Windows clients was limited to *LanMan* printing calls. This is the same protocol level as Windows 9x/Me PCs offer when they share printers. Beginning with the 2.2.0 release, Samba started to support the native Windows NT printing mechanisms. These are implemented via *MS-RPC* (RPC = *Remote Procedure Calls*). MS-RPCs use the *SPOOLSS* named pipe for all printing.

The additional functionality provided by the new SPOOLSS support includes:

- Support for downloading printer driver files to Windows 95/98/NT/2000 clients upon demand (*Point'n'Print*).
- Uploading of printer drivers via the Windows NT *Add Printer Wizard* (APW) or the *Imprints*¹ tool set.

¹<http://imprints.sourceforge.net/>

- Support for the native MS-RPC printing calls such as `StartDocPrinter`, `EnumJobs()`, and so on. (See the MSDN documentation² for more information on the Win32 printing API).
- Support for NT *Access Control Lists* (ACL) on printer objects.
- Improved support for printer queue manipulation through the use of internal databases for spooled job information (implemented by various `*.tdb` files).

A benefit of updating is that Samba-3 is able to publish its printers to Active Directory (or LDAP).

A fundamental difference exists between MS Windows NT print servers and Samba operation. Windows NT permits the installation of local printers that are not shared. This is an artifact of the fact that any Windows NT machine (server or client) may be used by a user as a workstation. Samba will publish all printers that are made available, either by default or by specific declaration via printer-specific shares.

Windows NT/200x/XP Professional clients do not have to use the standard SMB printer share; they can print directly to any printer on another Windows NT host using MS-RPC. This, of course, assumes that the client has the necessary privileges on the remote host that serves the printer resource. The default permissions assigned by Windows NT to a printer gives the Print permissions to the well-known *Everyone* group. (The older clients of type Windows 9x/Me can only print to shared printers).

17.5.1 Point'n'Print Client Drivers on Samba Servers

There is much confusion about what all this means. The question is often asked, “*Is it or is it not necessary for printer drivers to be installed on a Samba host in order to support printing from Windows clients?*” The answer to this is no, it is not necessary.

Windows NT/2000 clients can, of course, also run their APW to install drivers *locally* (which then connect to a Samba-served print queue). This is the same method used by Windows 9x/Me clients. (However, a *bug* existed in Samba 2.2.0 that made Windows NT/2000 clients require that the Samba server possess a valid driver for the printer. This was fixed in Samba 2.2.1).

But it is a new capability to install the printer drivers into the `[print$]` share of the Samba server, and a big convenience, too. Then *all* clients (including 95/98/ME) get the driver installed when they first connect to this printer share. The *uploading* or *depositing* of the driver into this `[print$]` share and the following binding of this driver to an existing Samba printer share can be achieved by different means:

- Running the *APW* on an NT/200x/XP Professional client (this does not work from 95/98/ME clients).
- Using the *Imprints* toolset.
- Using the *smclient* and *rpcclient* commandline tools.
- Using *cupsaddsmb* (only works for the CUPS printing system, not for LPR/LPD, LPRng, and so on).

²<http://msdn.microsoft.com/>

Samba does not use these uploaded drivers in any way to process spooled files. These drivers are utilized entirely by the clients who download and install them via the “*Point’n’Print*” mechanism supported by Samba. The clients use these drivers to generate print files in the format the printer (or the UNIX print system) requires. Print files received by Samba are handed over to the UNIX printing system, which is responsible for all further processing, as needed.

17.5.2 The Obsoleted `[printer$]` Section

Versions of Samba prior to 2.2 made it possible to use a share named `[printer$]`. This name was taken from the same named service created by Windows 9x/Me clients when a printer was shared by them. Windows 9x/Me printer servers always have a `[printer$]` service that provides read-only access (with no password required) to support printer driver downloads. However, Samba’s initial implementation allowed for a parameter named `printer driver location` to be used on a per share basis. This specified the location of the driver files associated with that printer. Another parameter named `printer driver` provided a means of defining the printer driver name to be sent to the client.

These parameters, including the `printer driver file` parameter, are now removed and cannot be used in installations of Samba-3. The share name `[print$]` is now used for the location of downloadable printer drivers. It is taken from the `[print$]` service created by Windows NT PCs when a printer is shared by them. Windows NT print servers always have a `[print$]` service that provides read-write access (in the context of its ACLs) to support printer driver downloads and uploads. This does not mean Windows 9x/Me clients are now thrown aside. They can use Samba’s `[print$]` share support just fine.

17.5.3 Creating the `[print$]` Share

In order to support the uploading and downloading of printer driver files, you must first configure a file share named `[print$]`. The public name of this share is hard coded in the MS Windows clients. It cannot be renamed since Windows clients are programmed to search for a service of exactly this name if they want to retrieve printer driver files.

You should modify the server’s file to add the global parameters and create the `[print$]` file share (of course, some of the parameter values, such as `path` are arbitrary and should be replaced with appropriate values for your site). See Example 17.3.

Of course, you also need to ensure that the directory named by the `path` parameter exists on the UNIX file system.

17.5.4 `[print$]` Section Parameters

The `[print$]` is a special section in `smb.conf`. It contains settings relevant to potential printer driver download and is used by windows clients for local print driver installation. The following parameters are frequently needed in this share section:

Example 17.3. [print\$] example

```

[global]
# members of the ntadmin group should be able to add drivers and set
# printer properties. root is implicitly always a 'printer admin'.
printer admin = @ntadmin
...

[printers]
...

[print$]
comment = Printer Driver Download Area
path = /etc/samba/drivers
browseable = yes
guest ok = yes
read only = yes
write list = @ntadmin, root

```

comment = **Printer Driver Download Area** — The comment appears next to the share name if it is listed in a share list (usually Windows clients will not see it, but it will also appear up in a **smbclient -L sambaserver** output).

path = **/etc/samba/printers** — Is the path to the location of the Windows driver file deposit from the UNIX point of view.

browseable = **no** — Makes the [print\$] share invisible to clients from the **Network Neighborhood**. However, you can still mount it from any client using the **net use g:\\sambaserver\print\$** command in a DOS-box or the **Connect network drive menu>** from Windows Explorer.

guest ok = **yes** — Gives read-only access to this share for all guest users. Access may be granted to download and install printer drivers on clients. The requirement for *guest ok* = **yes** depends on how your site is configured. If users will be guaranteed to have an account on the Samba host, then this is a non-issue.

NOTE



If all your Windows NT users are guaranteed to be authenticated by the Samba server (for example, if Samba authenticates via an NT domain server and the user has already been validated by the Domain Controller in order to logon to the Windows NT session), then guest access is not necessary. Of course, in a workgroup environment where you just want to print without worrying about silly accounts and security, then configure the share for guest access. You should consider adding *map to guest = Bad User* in the *[global]* section as well. Make sure you understand what this parameter does before using it.

read only = yes — Because we do not want everybody to upload driver files (or even change driver settings), we tagged this share as not writeable.

write list = @ntadmin, root — The *[print\$]* was made read-only by the previous setting so we should create a *write list* entry also. UNIX groups (denoted with a leading “@” character). Users listed here are allowed write-access (as an exception to the general public’s read-only access), which they need to update files on the share. Normally, you will want to only name administrative-level user account in this setting. Check the file system permissions to make sure these accounts can copy files to the share. If this is a non-root account, then the account should also be mentioned in the global *printer admin* parameter. See the *smb.conf* man page for more information on configuring file shares.

17.5.5 The *[print\$]* Share Directory

In order for a Windows NT print server to support the downloading of driver files by multiple client architectures, you must create several subdirectories within the *[print\$]* service (i.e., the UNIX directory named by the *path* parameter). These correspond to each of the supported client architectures. Samba follows this model as well. Just like the name of the *[print\$]* share itself, the subdirectories must be exactly the names listed below (you may leave out the subdirectories of architectures you do not need to support).

Therefore, create a directory tree below the *[print\$]* share for each architecture you wish to support like this:

```
[print$]--+
|--W32X86          # serves drivers to Windows NT x86
|--WIN40           # serves drivers to Windows 95/98
|--W32ALPHA       # serves drivers to Windows NT Alpha_AXP
|--W32MIPS        # serves drivers to Windows NT R4000
|--W32PPC         # serves drivers to Windows NT PowerPC
```

REQUIRED PERMISSIONS

In order to add a new driver to your Samba host, one of two conditions must hold true:



- The account used to connect to the Samba host must have a UID of 0 (i.e., a root account).
- The account used to connect to the Samba host must be named in the *printer adminlist*.

Of course, the connected account must still have write access to add files to the subdirectories beneath [print\$]. Remember that all file shares are set to “read-only” by default.

Once you have created the required [print\$] service and associated subdirectories, go to a Windows NT 4.0/200x/XP client workstation. Open **Network Neighborhood** or **My Network Places** and browse for the Samba host. Once you have located the server, navigate to its **Printers and Faxes** folder. You should see an initial listing of printers that matches the printer shares defined on your Samba host.

17.6 Installing Drivers into [print\$]

Have you successfully created the [print\$] share in `smb.conf`, and have you forced Samba to re-read its `smb.conf` file? Good. But you are not yet ready to use the new facility. The client driver files need to be installed into this share. So far it is still an empty share. Unfortunately, it is not enough to just copy the driver files over. They need to be correctly installed so that appropriate records for each driver will exist in the Samba internal databases so it can provide the correct drivers as they are requested from MS Windows clients. And that is a bit tricky, to say the least. We now discuss two alternative ways to install the drivers into [print\$]:

- Using the Samba commandline utility **rpcclient** with its various subcommands (here: **adddriver** and **setdriver**) from any UNIX workstation.
- Running a GUI (**Printer Properties** and **Add Printer Wizard**) from any Windows NT/200x/XP client workstation.

The latter option is probably the easier one (even if the process may seem a little bit weird at first).

17.6.1 Add Printer Wizard Driver Installation

The initial listing of printers in the Samba host’s **Printers** folder accessed from a client’s Explorer will have no real printer driver assigned to them. By default this driver name is set to a null string. This must be changed now. The local **Add Printer Wizard** (APW), run from NT/2000/XP clients, will help us in this task.

Installation of a valid printer driver is not straightforward. You must attempt to view the printer properties for the printer to which you want the driver assigned. Open the Windows Explorer, open **Network Neighborhood**, browse to the Samba host, open Samba's **Printers** folder, right-click on the printer icon and select **Properties...** You are now trying to view printer and driver properties for a queue that has this default NULL driver assigned. This will result in the following error message:

Device settings cannot be displayed. The driver for the specified printer is not installed, only spooler properties will be displayed. Do you want to install the driver now?

Do not click on **Yes!** Instead, click on **No** in the error dialog. Only now you will be presented with the printer properties window. From here, the way to assign a driver to a printer is open to us. You now have the choice of:

- Select a driver from the pop-up list of installed drivers. Initially this list will be empty.
- Click on **New Driver** to install a new printer driver (which will start up the APW).

Once the APW is started, the procedure is exactly the same as the one you are familiar with in Windows (we assume here that you are familiar with the printer driver installations procedure on Windows NT). Make sure your connection is, in fact, setup as a user with *printer admin* privileges (if in doubt, use **smbstatus** to check for this). If you wish to install printer drivers for client operating systems other than Windows NT x86, you will need to use the **Sharing** tab of the printer properties dialog.

Assuming you have connected with an administrative (or root) account (as named by the *printer admin* parameter), you will also be able to modify other printer properties such as ACLs and default device settings using this dialog. For the default device settings, please consider the advice given further in Section 17.6.2.

17.6.2 Installing Print Drivers Using **rpcclient**

The second way to install printer drivers into *[print\$]* and set them up in a valid way is to do it from the UNIX command line. This involves four distinct steps:

1. Gather info about required driver files and collect the files.
2. Deposit the driver files into the *[print\$]* share's correct subdirectories (possibly by using **smbclient**).
3. Run the **rpcclient** command line utility once with the **adddriver** subcommand.
4. Run **rpcclient** a second time with the **setdriver** subcommand.

We provide detailed hints for each of these steps in the paragraphs that follow.

17.6.2.1 Identifying Driver Files

To find out about the driver files, you have two options. You could check the contents of the driver CDROM that came with your printer. Study the **.inf* files located on the CDROM. This may not be possible, since the **.inf* file might be missing. Unfortunately, vendors have now started to use their own installation programs. These installations packages are often in some Windows platform archive format. Additionally, the files may be re-named

during the installation process. This makes it extremely difficult to identify the driver files required.

Then you only have the second option. Install the driver locally on a Windows client and investigate which file names and paths it uses after they are installed. (You need to repeat this procedure for every client platform you want to support. We show it here for the W32X86 platform only, a name used by Microsoft for all Windows NT/200x/XP clients.)

A good method to recognize the driver files is to print the test page from the driver's **Properties** dialog (**General** tab). Then look at the list of driver files named on the printout. You'll need to recognize what Windows (and Samba) are calling the **Driver File**, **Data File**, **Config File**, **Help File** and (optionally) the **Dependent Driver Files** (this may vary slightly for Windows NT). You need to take a note of all file names for the next steps.

Another method to quickly test the driver filenames and related paths is provided by the **rpcclient** utility. Run it with **enumdrivers** or with the **getdriver** subcommand, each at the 3 info level. In the following example, *TURBO_XP* is the name of the Windows PC (in this case it was a Windows XP Professional laptop). I installed the driver locally to *TURBO_XP*, from a Samba server called *KDE-BITSHOP*. We could run an interactive **rpcclient** session; then we would get an **rpcclient** /> prompt and would type the subcommands at this prompt. This is left as a good exercise to the reader. For now, we use **rpcclient** with the **-c** parameter to execute a single subcommand line and exit again. This is the method you would use if you want to create scripts to automate the procedure for a large number of printers and drivers. Note the different quotes used to overcome the different spaces in between words:

```
root# rpcclient -U'Danka%xxxx' -c \
    'getdriver "Heidelberg Digimaster 9110 (PS)" 3' TURBO_XP
cmd = getdriver "Heidelberg Digimaster 9110 (PS)" 3
```

[Windows NT x86]

Printer Driver Info 3:

```
Version: [2]
Driver Name: [Heidelberg Digimaster 9110 (PS)]
Architecture: [Windows NT x86]
Driver Path: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\HDNIS01_de.DLL]
Datafile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.ppd]
Configfile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\HDNIS01U_de.DLL]
Helpfile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\HDNIS01U_de.HLP]

Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.DLL]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.INI]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.dat]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.cat]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.def]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.hre]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.vnd]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\Hddm91c1_de.hlp]
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\HDNIS01Aux.dll]
```

```
Dependentfiles: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\HDNIS01_de.NTF]
```

```
Monitorname: []
```

```
Defaultdatatype: []
```

You may notice that this driver has quite a large number of **Dependent files** (there are worse cases, however). Also, strangely, the **Driver File** is tagged here **Driver Path**. We do not yet have support for the so-called WIN40 architecture installed. This name is used by Microsoft for the Windows 9x/Me platforms. If we want to support these, we need to install the Windows 9x/Me driver files in addition to those for W32X86 (i.e., the Windows NT72000/XP clients) onto a Windows PC. This PC can also host the Windows 9x/Me drivers, even if it runs on Windows NT, 2000 or XP.

Since the `[print$]` share is usually accessible through the **Network Neighborhood**, you can also use the UNC notation from Windows Explorer to poke at it. The Windows 9x/Me driver files will end up in subdirectory 0 of the WIN40 directory. The full path to access them will be `\\WINDOWSHOST\print$\WIN40\0\`.

NOTE



More recent drivers on Windows 2000 and Windows XP are installed into the “3” subdirectory instead of the “2”. The version 2 of drivers, as used in Windows NT, were running in Kernel Mode. Windows 2000 changed this. While it still can use the Kernel Mode drivers (if this is enabled by the Admin), its native mode for printer drivers is User Mode execution. This requires drivers designed for this. These types of drivers install into the “3” subdirectory.

17.6.2.2 Obtaining Driver Files from Windows Client [print\$] Shares

Now we need to collect all the driver files we identified in our previous step. Where do we get them from? Well, why not retrieve them from the very PC and the same `[print$]` share that we investigated in our last step to identify the files? We can use **smbclient** to do this. We will use the paths and names that were leaked to us by **getdriver**. The listing is edited to include linebreaks for readability:

```
root# smbclient //TURBO_XP/print\$ -U'Danka%xxxx' \
  -c 'cd W32X86/2;mget HD*_de.* hd*ppd Hd*_de.* Hddm*dll HDN*Aux.DLL'

added interface ip=10.160.51.60 bcast=10.160.51.255 nmask=255.255.252.0
Got a positive name query response from 10.160.50.8 ( 10.160.50.8 )
Domain=[DEVELOPMENT] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]
Get file Hddm91c1_de.ABD? n
Get file Hddm91c1_de.def? y
```



```
getting file \W32X86\2\Hddm91c1_de.def of size 428 as Hddm91c1_de.def
Get file Hddm91c1_de.DLL? y
getting file \W32X86\2\Hddm91c1_de.DLL of size 876544 as Hddm91c1_de.DLL
[...]
```

After this command is complete, the files are in our current local directory. You probably have noticed that this time we passed several commands to the `-c` parameter, separated by semi-colons. This effects that all commands are executed in sequence on the remote Windows server before `smbclient` exits again.

Remember to repeat the procedure for the WIN40 architecture should you need to support Windows 9x/Me/XP clients. Remember too, the files for these architectures are in the WIN40/0/ subdirectory. Once this is complete, we can run `smbclient ... put` to store the collected files on the Samba server's [print\$] share.

17.6.2.3 Installing Driver Files into [print\$]

We are now going to locate the driver files into the [print\$] share. Remember, the UNIX path to this share has been defined previously in your words missing here. You also have created subdirectories for the different Windows client types you want to support. Supposing your [print\$] share maps to the UNIX path `/etc/samba/drivers/`, your driver files should now go here:

- For all Windows NT, 2000 and XP clients into `/etc/samba/drivers/W32X86/` but not (yet) into the 2 subdirectory.
- For all Windows 95, 98 and ME clients into `/etc/samba/drivers/WIN40/` but not (yet) into the 0 subdirectory.

We again use `smbclient` to transfer the driver files across the network. We specify the same files and paths as were leaked to us by running `getdriver` against the original *Windows* install. However, now we are going to store the files into a *Samba/UNIX* print server's [print\$] share.

```
root# smbclient //SAMBA-CUPS/print\$\ -U'root%xxxx' -c \
'cd W32X86; put HDNIS01_de.DLL; \
put Hddm91c1_de.ppd; put HDNIS01U_de.DLL; \
put HDNIS01U_de.HLP; put Hddm91c1_de.DLL; \
put Hddm91c1_de.INI; put Hddm91c1KMMIn.DLL; \
put Hddm91c1_de.dat; put Hddm91c1_de.dat; \
put Hddm91c1_de.def; put Hddm91c1_de.hre; \
put Hddm91c1_de.vnd; put Hddm91c1_de.hlp; \
put Hddm91c1_de_reg.HLP; put HDNIS01Aux.dll; \
put HDNIS01_de.NTF'
```

```
added interface ip=10.160.51.60 bcast=10.160.51.255 nmask=255.255.252.0
Got a positive name query response from 10.160.51.162 ( 10.160.51.162 )
Domain=[CUPS-PRINT] OS=[UNIX] Server=[Samba 2.2.7a]
```

```

putting file HDNIS01_de.DLL as \W32X86\HDNIS01_de.DLL
putting file Hddm91c1_de.ppd as \W32X86\Hddm91c1_de.ppd
putting file HDNIS01U_de.DLL as \W32X86\HDNIS01U_de.DLL
putting file HDNIS01U_de.HLP as \W32X86\HDNIS01U_de.HLP
putting file Hddm91c1_de.DLL as \W32X86\Hddm91c1_de.DLL
putting file Hddm91c1_de.INI as \W32X86\Hddm91c1_de.INI
putting file Hddm91c1KMMIn.DLL as \W32X86\Hddm91c1KMMIn.DLL
putting file Hddm91c1_de.dat as \W32X86\Hddm91c1_de.dat
putting file Hddm91c1_de.dat as \W32X86\Hddm91c1_de.dat
putting file Hddm91c1_de.def as \W32X86\Hddm91c1_de.def
putting file Hddm91c1_de.hre as \W32X86\Hddm91c1_de.hre
putting file Hddm91c1_de.vnd as \W32X86\Hddm91c1_de.vnd
putting file Hddm91c1_de.hlp as \W32X86\Hddm91c1_de.hlp
putting file Hddm91c1_de_reg.HLP as \W32X86\Hddm91c1_de_reg.HLP
putting file HDNIS01Aux.dll as \W32X86\HDNIS01Aux.dll
putting file HDNIS01_de.NTF as \W32X86\HDNIS01_de.NTF

```

Whew — that was a lot of typing! Most drivers are a lot smaller — many only having three generic PostScript driver files plus one PPD. While we did retrieve the files from the 2 subdirectory of the W32X86 directory from the Windows box, we do not put them (for now) in this same subdirectory of the Samba box. This relocation will automatically be done by the **adddriver** command, which we will run shortly (and do not forget to also put the files for the Windows 9x/Me architecture into the WIN40/ subdirectory should you need them).

17.6.2.4 smbclient to Confirm Driver Installation

For now we verify that our files are there. This can be done with **smbclient**, too (but, of course, you can log in via SSH also and do this through a standard UNIX shell access):

```

root# smbclient //SAMBA-CUPS/print$$$ -U 'root%xxxx' \
  -c 'cd W32X86; pwd; dir; cd 2; pwd; dir'
added interface ip=10.160.51.60 bcast=10.160.51.255 nmask=255.255.252.0
Got a positive name query response from 10.160.51.162 ( 10.160.51.162 )
Domain=[CUPS-PRINT] OS=[UNIX] Server=[Samba 2.2.8a]

```

```

Current directory is \\SAMBA-CUPS\print$\W32X86\
.                D            0  Sun May  4 03:56:35 2003
..               D            0  Thu Apr 10 23:47:40 2003
2                D            0  Sun May  4 03:56:18 2003
HDNIS01Aux.dll   A       15356  Sun May  4 03:58:59 2003
Hddm91c1KMMIn.DLL A       46966  Sun May  4 03:58:59 2003
HDNIS01_de.DLL  A      434400  Sun May  4 03:58:59 2003
HDNIS01_de.NTF  A       790404 Sun May  4 03:56:35 2003
Hddm91c1_de.DLL A      876544  Sun May  4 03:58:59 2003
Hddm91c1_de.INI A           101 Sun May  4 03:58:59 2003
Hddm91c1_de.dat A        5044  Sun May  4 03:58:59 2003

```

```

Hddm91c1_de.def          A      428  Sun May  4 03:58:59 2003
Hddm91c1_de.hlp          A     37699 Sun May  4 03:58:59 2003
Hddm91c1_de.hre          A    323584 Sun May  4 03:58:59 2003
Hddm91c1_de.ppd          A     26373 Sun May  4 03:58:59 2003
Hddm91c1_de.vnd          A     45056 Sun May  4 03:58:59 2003
HDNIS01U_de.DLL          A    165888 Sun May  4 03:58:59 2003
HDNIS01U_de.HLP          A     19770 Sun May  4 03:58:59 2003
Hddm91c1_de_reg.HLP      A    228417 Sun May  4 03:58:59 2003
40976 blocks of size 262144. 709 blocks available

```

```

Current directory is \\SAMBA-CUPS\print$\W32X86\2\
.                D         0  Sun May  4 03:56:18 2003
..               D         0  Sun May  4 03:56:35 2003
ADOBEPS5.DLL     A    434400 Sat May  3 23:18:45 2003
laserjet4.ppd    A     9639  Thu Apr 24 01:05:32 2003
ADOBEPSU.DLL     A    109568 Sat May  3 23:18:45 2003
ADOBEPSU.HLP     A     18082 Sat May  3 23:18:45 2003
PDFcreator2.PPD  A     15746 Sun Apr 20 22:24:07 2003
40976 blocks of size 262144. 709 blocks available

```

Notice that there are already driver files present in the 2 subdirectory (probably from a previous installation). Once the files for the new driver are there too, you are still a few steps away from being able to use them on the clients. The only thing you could do now is to retrieve them from a client just like you retrieve ordinary files from a file share, by opening `print$` in Windows Explorer. But that wouldn't install them per Point'n'Print. The reason is: Samba does not yet know that these files are something special, namely *printer driver files* and it does not know to which print queue(s) these driver files belong.

17.6.2.5 Running rpcclient with adddriver

Next, you must tell Samba about the special category of the files you just uploaded into the `[print$]` share. This is done by the `adddriver` command. It will prompt Samba to register the driver files into its internal TDB database files. The following command and its output has been edited, again, for readability:

```

root# rpcclient -Uroot%xxxx -c 'adddriver "Windows NT x86" \
  "dm9110:HDNIS01_de.DLL: \
  Hddm91c1_de.ppd:HDNIS01U_de.DLL:HDNIS01U_de.HLP: \
  NULL:RAW:Hddm91c1_de.DLL,Hddm91c1_de.INI, \
  Hddm91c1_de.dat,Hddm91c1_de.def,Hddm91c1_de.hre, \
  Hddm91c1_de.vnd,Hddm91c1_de.hlp,Hddm91c1KMMIn.DLL, \
  HDNIS01Aux.dll,HDNIS01_de.NTF, \
  Hddm91c1_de_reg.HLP' SAMBA-CUPS

cmd = adddriver "Windows NT x86" \
  "dm9110:HDNIS01_de.DLL:Hddm91c1_de.ppd:HDNIS01U_de.DLL: \

```

```

HDNIS01U_de.HLP:NULL:RAW:Hddm91c1_de.DLL,Hddm91c1_de.INI, \
Hddm91c1_de.dat,Hddm91c1_de.def,Hddm91c1_de.hre, \
Hddm91c1_de.vnd,Hddm91c1_de.hlp,Hddm91c1KMMin.DLL, \
HDNIS01Aux.dll,HDNIS01_de.NTF,Hddm91c1_de_reg.HLP"

```

Printer Driver dm9110 successfully installed.

After this step, the driver should be recognized by Samba on the print server. You need to be very careful when typing the command. Don't exchange the order of the fields. Some changes would lead to an `NT_STATUS_UNSUCCESSFUL` error message. These become obvious. Other changes might install the driver files successfully, but render the driver unworkable. So take care! Hints about the syntax of the `adddriver` command are in the man page. The CUPS printing chapter provides a more detailed description, should you need it.

17.6.2.6 Checking adddriver Completion

One indication for Samba's recognition of the files as driver files is the `successfully installed` message. Another one is the fact that our files have been moved by the `adddriver` command into the 2 subdirectory. You can check this again with `smbclient`:

```

root# smbclient //SAMBA-CUPS/print\$ -Uroot%xx \
  -c 'cd W32X86;dir;pwd;cd 2;dir;pwd'
added interface ip=10.160.51.162 bcast=10.160.51.255 nmask=255.255.252.0
Domain=[CUPS-PRINT] OS=[UNIX] Server=[Samba 2.2.7a]

Current directory is \\SAMBA-CUPS\print\$W32X86\
.                               D           0   Sun May  4 04:32:48 2003
..                              D           0   Thu Apr 10 23:47:40 2003
2                               D           0   Sun May  4 04:32:48 2003
                                40976 blocks of size 262144. 731 blocks available

Current directory is \\SAMBA-CUPS\print\$W32X86\2\
.                               D           0   Sun May  4 04:32:48 2003
..                              D           0   Sun May  4 04:32:48 2003
DigiMaster.PPD                 A    148336  Thu Apr 24 01:07:00 2003
ADOBEPS5.DLL                   A    434400  Sat May  3 23:18:45 2003
laserjet4.ppd                  A     9639  Thu Apr 24 01:05:32 2003
ADOBEPSU.DLL                   A    109568  Sat May  3 23:18:45 2003
ADOBEPSU.HLP                   A     18082  Sat May  3 23:18:45 2003
PDFcreator2.PPD                A     15746  Sun Apr 20 22:24:07 2003
HDNIS01Aux.dll                 A     15356  Sun May  4 04:32:18 2003
Hddm91c1KMMin.DLL              A     46966  Sun May  4 04:32:18 2003
HDNIS01_de.DLL                 A    434400  Sun May  4 04:32:18 2003
HDNIS01_de.NTF                 A    790404  Sun May  4 04:32:18 2003
Hddm91c1_de.DLL                A    876544  Sun May  4 04:32:18 2003
Hddm91c1_de.INI                A       101  Sun May  4 04:32:18 2003

```

```

Hddm91c1_de.dat           A      5044  Sun May  4 04:32:18 2003
Hddm91c1_de.def           A       428  Sun May  4 04:32:18 2003
Hddm91c1_de.hlp           A     37699  Sun May  4 04:32:18 2003
Hddm91c1_de.hre           A    323584  Sun May  4 04:32:18 2003
Hddm91c1_de.ppd           A     26373  Sun May  4 04:32:18 2003
Hddm91c1_de.vnd           A     45056  Sun May  4 04:32:18 2003
HDNIS01U_de.DLL           A    165888  Sun May  4 04:32:18 2003
HDNIS01U_de.HLP           A     19770  Sun May  4 04:32:18 2003
Hddm91c1_de_reg.HLP       A    228417  Sun May  4 04:32:18 2003
40976 blocks of size 262144. 731 blocks available

```

Another verification is that the timestamp of the printing TDB files is now updated (and possibly their file size has increased).

17.6.2.7 Check Samba for Driver Recognition

Now the driver should be registered with Samba. We can easily verify this, and will do so in a moment. However, this driver is not yet associated with a particular printer. We may check the driver status of the files by at least three methods:

- From any Windows client browse Network Neighborhood, find the Samba host and open the Samba **Printers and Faxes** folder. Select any printer icon, right-click and select the printer **Properties**. Click the **Advanced** tab. Here is a field indicating the driver for that printer. A drop-down menu allows you to change that driver (be careful not to do this unwittingly). You can use this list to view all drivers known to Samba. Your new one should be among them. (Each type of client will only see his own architecture's list. If you do not have every driver installed for each platform, the list will differ if you look at it from Windows95/98/ME or WindowsNT/2000/XP.)
- From a Windows 200x/XP client (not Windows NT) browse **Network Neighborhood**, search for the Samba server and open the server's **Printers** folder, right-click on the white background (with no printer highlighted). Select **Server Properties**. On the **Drivers** tab you will see the new driver listed. This view enables you to also inspect the list of files belonging to that driver (this does not work on Windows NT, but only on Windows 2000 and Windows XP; Windows NT does not provide the **Drivers** tab). An alternative and much quicker method for Windows 2000/XP to start this dialog is by typing into a DOS box (you must of course adapt the name to your Samba server instead of *SAMBA-CUPS*):

```
rundll32 printui.dll,PrintUIEntry /s /t2 /n\\SAMBA-CUPS
```

- From a UNIX prompt, run this command (or a variant thereof) where *SAMBA-CUPS* is the name of the Samba host and *xxxx* represents the actual Samba password assigned to root:

```
rpcclient -U'root%xxxx' -c 'enumdrivers' SAMBA-CUPS
```

You will see a listing of all drivers Samba knows about. Your new one should be among them. But it is only listed under the *[Windows NT x86]* heading, not under *[Windows 4.0]*, since you didn't install that part. Or did you? You will see a listing

of all drivers Samba knows about. Your new one should be among them. In our example it is named `dm9110`. Note that the third column shows the other installed drivers twice, one time for each supported architecture. Our new driver only shows up for Windows NT 4.0 or 2000. To have it present for Windows 95, 98 and ME, you'll have to repeat the whole procedure with the WIN40 architecture and subdirectory.

17.6.2.8 Specific Driver Name Flexibility

You can name the driver as you like. If you repeat the **adddriver** step with the same files as before but with a different driver name, it will work the same:

```
root# rpcclient -Uroot%xxxx \
  -c 'adddriver "Windows NT x86" \
  "mydrivername:HDNIS01_de.DLL: \
  Hddm91c1_de.ppd:HDNIS01U_de.DLL:HDNIS01U_de.HLP: \
  NULL:RAW:Hddm91c1_de.DLL,Hddm91c1_de.INI, \
  Hddm91c1_de.dat,Hddm91c1_de.def,Hddm91c1_de.hre, \
  Hddm91c1_de.vnd,Hddm91c1_de.hlp,Hddm91c1KMMin.DLL, \
  HDNIS01Aux.dll,HDNIS01_de.NTF,Hddm91c1_de_reg.HLP' SAMBA-CUPS
```

```
cmd = adddriver "Windows NT x86" \
  "mydrivername:HDNIS01_de.DLL:Hddm91c1_de.ppd:HDNIS01U_de.DLL:\
  HDNIS01U_de.HLP:NULL:RAW:Hddm91c1_de.DLL,Hddm91c1_de.INI, \
  Hddm91c1_de.dat,Hddm91c1_de.def,Hddm91c1_de.hre, \
  Hddm91c1_de.vnd,Hddm91c1_de.hlp,Hddm91c1KMMin.DLL, \
  HDNIS01Aux.dll,HDNIS01_de.NTF,Hddm91c1_de_reg.HLP"
```

Printer Driver `mydrivername` successfully installed.

You will be able to bind that driver to any print queue (however, you are responsible that you associate drivers to queues that make sense with respect to target printers). You cannot run the **rpcclient adddriver** command repeatedly. Each run consumes the files you had put into the `[print$]` share by moving them into the respective subdirectories. So you must execute an **smbclient ... put** command before each **rpcclient ... adddriver** command.

17.6.2.9 Running rpcclient with the setdriver

Samba needs to know which printer owns which driver. Create a mapping of the driver to a printer, and store this info in Samba's memory, the TDB files. The **rpcclient setdriver** command achieves exactly this:

```
root# rpcclient -U'root%xxxx' -c 'setdriver dm9110 mydrivername' SAMBA-CUPS
cmd = setdriver dm9110 mydrivername
```

Successfully set dm9110 to driver mydrivername.

Ah, no, I did not want to do that. Repeat, this time with the name I intended:

```
root# rpcclient -U'root%xxxx' -c 'setdriver dm9110 dm9110' SAMBA-CUPS
  cmd = setdriver dm9110 dm9110
Successfully set dm9110 to driver dm9110.
```

The syntax of the command is:

```
rpcclient -U'root%sambapassword' -c 'setdriver printername \
  drivename' SAMBA-Hostname.
```

Now we have done most of the work, but not all of it.

NOTE



The **setdriver** command will only succeed if the printer is already known to Samba. A bug in 2.2.x prevented Samba from recognizing freshly installed printers. You had to restart Samba, or at least send an HUP signal to all running `smbd` processes to work around this: `kill -HUP $(pidof smbd)`.

17.7 Client Driver Installation Procedure

As Don Quixote said: “*The proof of the pudding is in the eating.*” The proof for our setup lies in the printing. So let’s install the printer driver onto the client PCs. This is not as straightforward as it may seem. Read on.

17.7.1 First Client Driver Installation

Especially important is the installation onto the first client PC (for each architectural platform separately). Once this is done correctly, all further clients are easy to setup and shouldn’t need further attention. What follows is a description for the recommended first procedure. You work now from a client workstation. You should guarantee that your connection is not unwittingly mapped to *bad user* nobody. In a DOS box type:

```
net use \\SAMBA-SERVER\print$ /user:root
```

Replace root, if needed, by another valid *printer admin* user as given in the definition. Should you already be connected as a different user, you will get an error message. There is no easy way to get rid of that connection, because Windows does not seem to know a

concept of logging off from a share connection (do not confuse this with logging off from the local workstation; that is a different matter). You can try to close all Windows file explorer and Internet Explorer for Windows. As a last resort, you may have to reboot. Make sure there is no automatic reconnection set up. It may be easier to go to a different workstation and try from there. After you have made sure you are connected as a printer admin user (you can check this with the `smbstatus` command on Samba), do this from the Windows workstation:

1. Open **Network Neighborhood**.
2. Browse to Samba server.
3. Open its **Printers and Faxes** folder.
4. Highlight and right-click on the printer.
5. Select **Connect** (for Windows NT4/200x it is possibly **Install**).

A new printer (named *printername* on Samba-server) should now have appeared in your *local* Printer folder (check **Start – Settings – Control Panel – Printers and Faxes**).

Most likely you are now tempted to try to print a test page. After all, you now can open the printer properties, and on the **General** tab there is a button offering to do just that. But chances are that you get an error message saying Unable to print Test Page. The reason might be that there is not yet a valid Device Mode set for the driver, or that the “*Printer Driver Data*” set is still incomplete.

You must make sure that a valid *Device Mode* is set for the driver. We now explain what that means.

17.7.2 Setting Device Modes on New Printers

For a printer to be truly usable by a Windows NT/200x/XP client, it must possess:

- A valid *Device Mode* generated by the driver for the printer (defining things like paper size, orientation and duplex settings).
- A complete set of *Printer Driver Data* generated by the driver.

If either of these is incomplete, the clients can produce less than optimal output at best. In the worst cases, unreadable garbage or nothing at all comes from the printer or it produces a harvest of error messages when attempting to print. Samba stores the named values and all printing related information in its internal TDB database files (`ntprinters.tdb`, `ntdrivers.tdb`, `printing.tdb` and `ntforms.tdb`).

What do these two words stand for? Basically, the Device Mode and the set of Printer Driver Data is a collection of settings for all print queue properties, initialized in a sensible way. Device Modes and Printer Driver Data should initially be set on the print server (the Samba host) to healthy values so the clients can start to use them immediately. How do we set these initial healthy values? This can be achieved by accessing the drivers remotely from an NT (or 200x/XP) client, as is discussed in the following paragraphs.

Be aware that a valid Device Mode can only be initiated by a *printer admin*, or root (the reason should be obvious). Device Modes can only be correctly set by executing the printer

driver program itself. Since Samba cannot execute this Win32 platform driver code, it sets this field initially to NULL (which is not a valid setting for clients to use). Fortunately, most drivers automatically generate the Printer Driver Data that is needed when they are uploaded to the *[print\$]* share with the help of the APW or rpcclient.

The generation and setting of a first valid Device Mode, however, requires some tickling from a client, to set it on the Samba server. The easiest means of doing so is to simply change the page orientation on the server's printer. This executes enough of the printer driver program on the client for the desired effect to happen, and feeds back the new Device Mode to our Samba server. You can use the native Windows NT/200x/XP printer properties page from a Window client for this:

1. Browse the **Network Neighborhood**.
2. Find the Samba server.
3. Open the Samba server's **Printers and Faxes** folder.
4. Highlight the shared printer in question.
5. Right-click on the printer (you may already be here, if you followed the last section's description).
6. At the bottom of the context menu select **Properties** (if the menu still offers the **Connect** entry further above, you need to click on that one first to achieve the driver installation as shown in the last section).
7. Go to the **Advanced** tab; click on **Printing Defaults**.
8. Change the **Portrait** page setting to **Landscape** (and back).
9. Make sure to apply changes between swapping the page orientation to cause the change to actually take effect.
10. While you are at it, you may also want to set the desired printing defaults here, which then apply to all future client driver installations on the remaining from now on.

This procedure has executed the printer driver program on the client platform and fed back the correct Device Mode to Samba, which now stored it in its TDB files. Once the driver is installed on the client, you can follow the analogous steps by accessing the *local* **Printers** folder, too, if you are a Samba printer admin user. From now on, printing should work as expected.

Samba includes a service level parameter name *default devmode* for generating a default Device Mode for a printer. Some drivers will function well with Samba's default set of properties. Others may crash the client's spooler service. So use this parameter with caution. It is always better to have the client generate a valid device mode for the printer and store it on the server for you.

17.7.3 Additional Client Driver Installation

Every additional driver may be installed, along the lines described above. Browse network, open the **Printers** folder on Samba server, right-click on **Printer** and choose **Connect....** Once this completes (should be not more than a few seconds, but could also take a minute,

depending on network conditions), you should find the new printer in your client workstation local **Printers and Faxes** folder.

You can also open your local **Printers and Faxes** folder by using this command on Windows 200x/XP Professional workstations:

```
rundll32 shell32.dll,SHHelpShortcuts_RunDLL PrintersFolder
```

or this command on Windows NT 4.0 workstations:

```
rundll32 shell32.dll,Control_RunDLL MAIN.CPL @2
```

You can enter the commands either inside a **DOS box** window or in the **Run command...** field from the **Start** menu.

17.7.4 Always Make First Client Connection as root or “*printer admin*”

After you installed the driver on the Samba server (in its [*print\$*] share, you should always make sure that your first client installation completes correctly. Make it a habit for yourself to build the very first connection from a client as *printer admin*. This is to make sure that:

- A first valid *Device Mode* is really initialized (see above for more explanation details).
- The default print settings of your printer for all further client installations are as you want them.

Do this by changing the orientation to landscape, click on **Apply**, and then change it back again. Next, modify the other settings (for example, you do not want the default media size set to **Letter** when you are all using **A4**, right? You may want to set the printer for **duplex** as the default, and so on).

To connect as root to a Samba printer, try this command from a Windows 200x/XP DOS box command prompt:

```
C:\> runas /netonly /user:root "rundll32 printui.dll,PrintUIEntry /p /t3 /n
  \\SAMBA-SERVER\printername"
```

You will be prompted for root’s Samba-password; type it, wait a few seconds, click on **Printing Defaults**, and proceed to set the job options that should be used as defaults by all clients. Alternately, instead of root you can name one other member of the *printer admin* from the setting.

Now all the other users downloading and installing the driver the same way (called “*Point’n’Print*”) will have the same defaults set for them. If you miss this step you’ll get a lot of Help Desk calls from your users, but maybe you like to talk to people.

17.8 Other Gotchas

Your driver is installed. It is now ready for Point’n’Print installation by the clients. You may have tried to download and use it onto your first client machine, but wait. Let’s make

sure you are acquainted first with a few tips and tricks you may find useful. For example, suppose you did not set the defaults on the printer, as advised in the preceding paragraphs. Your users complain about various issues (such as, “*We need to set the paper size for each job from Letter to A4 and it will not store it.*”)

17.8.1 Setting Default Print Options for Client Drivers

The last sentence might be viewed with mixed feelings by some users and admins. They have struggled for hours and could not arrive at a point where their settings seemed to be saved. It is not their fault. The confusing thing is that in the multi-tabbed dialog that pops up when you right-click on the printer name and select **Properties**, you can arrive at two dialogs that appear identical, each claiming that they help you to set printer options in three different ways. Here is the definite answer to the Samba default driver setting FAQ:

How are you doing it? I bet the wrong way. (It is not easy to find out, though). There are three different ways to bring you to a dialog that seems to set everything. All three dialogs look the same, but only one of them does what you intend. You need to be Administrator or Print Administrator to do this for all users. Here is how I reproduce it in an XP Professional:

A The first “*wrong*” way:

- 1 Open the **Printers** folder.
- 2 Right-click on the printer (*remotepri nter on cupshost*) and select in context menu **Printing Preferences...**
- 3 Look at this dialog closely and remember what it looks like.

B The second “*wrong*” way:

- 1 Open the **Printers** folder.
- 2 Right-click on the printer (*remotepri nter on cupshost*) and select in the context menu **Properties**
- 3 Click on the **General** tab
- 4 Click on the **Printing Preferences...**
- 5 A new dialog opens. Keep this dialog open and go back to the parent dialog.

C The third and correct way: (should you do this from the beginning, just carry out steps 1 and 2 from the second method above).

- 1 Click on the **Advanced** tab. (If everything is “*grayed out*,” then you are not logged in as a user with enough privileges).
- 2 Click on the **Printing Defaults** button.
- 3 On any of the two new tabs, click on the **Advanced** button.
- 4 A new dialog opens. Compare this one to the other. Are they identical looking comparing one from “*B.5*” and one from *A.3*”.

Do you see any difference in the two settings dialogs? I do not either. However, only the last one, which you arrived at with steps C.1 through 6 will permanently save any settings

which will then become the defaults for new users. If you want all clients to have the same defaults, you need to conduct these steps as administrator (*printer admin* in) before a client downloads the driver (the clients can later set their own per-user defaults by following procedures A or B above). Windows 200x/XP allow per-user default settings and the ones the administrator gives them, before they set up their own. The parents of the identically-looking dialogs have a slight difference in their window names; one is called **Default Print Values for Printer Foo on Server Bar**" (which is the one you need) and the other is called "*Print Settings for Printer Foo on Server Bar*". The last one is the one you arrive at when you right-click on the printer and select **Print Settings...** This is the one that you were taught to use back in the days of Windows NT, so it is only natural to try the same way with Windows 200x/XP. You would not dream that there is now a different path to arrive at an identically looking, but functionally different, dialog to set defaults for all users.

TIP

Try (on Windows 200x/XP) to run this command (as a user with the right privileges):

```
rundll32 printui.dll,PrintUIEntry /p /t3 /
n\\SAMBA-SERVER\printersharename
```



To see the tab with the **Printing Defaults** button (the one you need),also run this command:

```
rundll32 printui.dll,PrintUIEntry /p /t0 /
n\\SAMBA-SERVER\printersharename
```

To see the tab with the **Printing Preferences** button (the one which does not set system-wide defaults), you can start the commands from inside a DOS box" or from **Start -> Run**.

17.8.2 Supporting Large Numbers of Printers

One issue that has arisen during the recent development phase of Samba is the need to support driver downloads for hunderds of printers. Using Windows NT APW here is somewhat awkward (to say the least). If you do not want to acquire RSS pains from the printer installation clicking orgy alone, you need to think about a non-interactive script.

If more than one printer is using the same driver, the **rpcclient setdriver** command can be used to set the driver associated with an installed queue. If the driver is uploaded to *[print\$]* once and registered with the printing TDBs, it can be used by multiple print queues. In this case, you just need to repeat the **setprinter** subcommand of **rpcclient** for every queue (without the need to conduct the **adddriver** repeatedly). The following is an example of how this could be accomplished:

```
root# rpcclient SAMBA-CUPS -U root%secret -c 'enumdrivers'
```

```

cmd = enumdrivers

[Windows NT x86]
Printer Driver Info 1:
  Driver Name: [infotec IS 2075 PCL 6]

Printer Driver Info 1:
  Driver Name: [DANKA InfoStream]

Printer Driver Info 1:
  Driver Name: [Heidelberg Digimaster 9110 (PS)]

Printer Driver Info 1:
  Driver Name: [dm9110]

Printer Driver Info 1:
  Driver Name: [mydrivername]

[....]

```

```

root# rpcclient SAMBA-CUPS -U root%secret -c 'enumprinters'
cmd = enumprinters
  flags:[0x800000]
  name:[\\SAMBA-CUPS\dm9110]
  description:[\\SAMBA-CUPS\dm9110,,110ppm HiVolume DANKA Stuttgart]
  comment:[110 ppm HiVolume DANKA Stuttgart]
[....]

```

```

root# rpcclient SAMBA-CUPS -U root%secret -c \
'setdriver dm9110 "Heidelberg Digimaster 9110 (PS)"'
cmd = setdriver dm9110 Heidelberg Digimaster 9110 (PPD)
Successfully set dm9110 to driver Heidelberg Digimaster 9110 (PS).

```

```

root# rpcclient SAMBA-CUPS -U root%secret -c 'enumprinters'
cmd = enumprinters
  flags:[0x800000]
  name:[\\SAMBA-CUPS\dm9110]
  description:[\\SAMBA-CUPS\dm9110,Heidelberg Digimaster 9110 (PS),\
  110ppm HiVolume DANKA Stuttgart]
  comment:[110ppm HiVolume DANKA Stuttgart]
[....]

```

```
root# rpcclient SAMBA-CUPS -U root%secret -c 'setdriver dm9110 mydrivername'
cmd = setdriver dm9110 mydrivername
Successfully set dm9110 to mydrivername.
```

```
root# rpcclient SAMBA-CUPS -U root%secret -c 'enumprinters'
cmd = enumprinters
flags:[0x800000]
name:[\\SAMBA-CUPS\dm9110]
description:[\\SAMBA-CUPS\dm9110,mydrivername,\
110ppm HiVolume DANKA Stuttgart]
comment:[110ppm HiVolume DANKA Stuttgart]
[....]
```

It may not be easy to recognize that the first call to **enumprinters** showed the “*dm9110*” printer with an empty string where the driver should have been listed (between the 2 commas in the description field). After the **setdriver** command succeeded, all is well.

17.8.3 Adding New Printers with the Windows NT APW

By default, Samba exhibits all printer shares defined in `smb.conf` in the **Printers** folder. Also located in this folder is the Windows NT Add Printer Wizard icon. The APW will be shown only if:

- The connected user is able to successfully execute an **OpenPrinterEx(\\server)** with administrative privileges (i.e., root or *printer admin*).

TIP



Try this from a Windows 200x/XP DOS box command prompt:

```
runas /netonly /user:root rundll32
printui.dll,PrintUIEntry /p /t0 /n
\\SAMBA-SERVER\printersharename
```

Click on **Printing Preferences**.

- ... contains the setting `show add printer wizard = yes` (the default).

The APW can do various things:

- Upload a new driver to the Samba `[print$]` share.
- Associate an uploaded driver with an existing (but still driverless) print queue.
- Exchange the currently used driver for an existing print queue with one that has been uploaded before.

- Add an entirely new printer to the Samba host (only in conjunction with a working *add printer command*. A corresponding *delete printer command* for removing entries from the **Printers** folder may also be provided).

The last one (add a new printer) requires more effort than the previous ones. To use the APW to successfully add a printer to a Samba server, the *add printer command* must have a defined value. The program hook must successfully add the printer to the UNIX print system (i.e., to `/etc/printcap`, `/etc/cups/printers.conf` or other appropriate files) and to `smb.conf` if necessary.

When using the APW from a client, if the named printer share does not exist, `smbd` will execute the *add printer command* and reparse to the to attempt to locate the new printer share. If the share is still not defined, an error of Access Denied is returned to the client. The *add printer command* is executed under the context of the connected user, not necessarily a root account. A *map to guest* = bad user may have connected you unwittingly under the wrong privilege. You should check it by using the `smbstatus` command.

17.8.4 Error Message: “Cannot connect under a different Name”

Once you are connected with the wrong credentials, there is no means to reverse the situation other than to close all Explorer Windows, and perhaps reboot.

- The `net use \\SAMBA-SERVER\sharename /user:root` gives you an error message: “Multiple connections to a server or a shared resource by the same user utilizing the several user names are not allowed. Disconnect all previous connections to the server, resp. the shared resource, and try again.”
- Every attempt to “connect a network drive” to `\\SAMBASERVER\print$` to `z:` is countered by the pertinacious message: “This network folder is currently connected under different credentials (username and password). Disconnect first any existing connection to this network share in order to connect again under a different username and password”.

So you close all connections. You try again. You get the same message. You check from the Samba side, using `smbstatus`. Yes, there are more connections. You kill them all. The client still gives you the same error message. You watch the `smbd.log` file on a high debug level and try reconnect. Same error message, but not a single line in the log. You start to wonder if there was a connection attempt at all. You run `ethereal` and `tcpdump` while you try to connect. Result: not a single byte goes on the wire. Windows still gives the error message. You close all Explorer windows and start it again. You try to connect — and this times it works! Windows seems to cache connection information somewhere and does not keep it up-to-date (if you are unlucky you might need to reboot to get rid of the error message).

17.8.5 Take Care When Assembling Driver Files

You need to be extremely careful when you take notes about the files and belonging to a particular driver. Don’t confuse the files for driver version “0” (for Windows 9x/Me, going into `[print$]/WIN/0/`), driver version 2 (Kernel Mode driver for Windows NT, going into `[print$]/W32X86/2/` may be used on Windows 200x/XP also), and driver version “3”

(non-Kernel Mode driver going into [print\$]/W32X86/3/ cannot be used on Windows NT). Quite often these different driver versions contain files that have the same name but actually are very different. If you look at them from the Windows Explorer (they reside in %WINDOWS%\system32\spool\drivers\W32X86\), you will probably see names in capital letters, while an **enumdrivers** command from Samba would show mixed or lower case letters. So it is easy to confuse them. If you install them manually using **rpcclient** and subcommands, you may even succeed without an error message. Only later, when you try install on a client, you will encounter error messages like **This server has no appropriate driver for the printer.**

Here is an example. You are invited to look closely at the various files, compare their names and their spelling, and discover the differences in the composition of the version 2 and 3 sets. Note: the version 0 set contained 40 *Dependentfiles*, so I left it out for space reasons:

```
root# rpcclient -U 'Administrator%secret' -c 'enumdrivers 3' 10.160.50.8
```

```
Printer Driver Info 3:
```

```
Version: [3]
```

```
Driver Name: [Canon iR8500 PS3]
```

```
Architecture: [Windows NT x86]
```

```
Driver Path: [\\10.160.50.8\print$\W32X86\3\cns3g.dll]
```

```
Datafile: [\\10.160.50.8\print$\W32X86\3\iR8500sg.xpd]
```

```
Configfile: [\\10.160.50.8\print$\W32X86\3\cns3gui.dll]
```

```
Helpfile: [\\10.160.50.8\print$\W32X86\3\cns3g.hlp]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\aucplmNT.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\ucs32p.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\tn132.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\aussdrv.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cnspdc.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\aussapi.dat]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cns3407.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\CnS3G.cnt]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\NBAPI.DLL]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\NBIPC.DLL]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcview.exe]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcdsp1.exe]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcedit.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcqm.exe]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcspl.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cfine32.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcr407.dll]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\Cpcqm407.hlp]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cpcqm407.cnt]
```

```
Dependentfiles: [\\10.160.50.8\print$\W32X86\3\cns3ggr.dll]
```

```
Monitorname: []
```

```
Defaultdatatype: []
```



```

Printer Driver Info 3:
  Version: [2]
  Driver Name: [Canon iR5000-6000 PS3]
  Architecture: [Windows NT x86]
  Driver Path: [\\10.160.50.8\print$\W32X86\2\cns3g.dll]
  Datafile: [\\10.160.50.8\print$\W32X86\2\IR5000sg.xpd]
  Configfile: [\\10.160.50.8\print$\W32X86\2\cns3gui.dll]
  Helpfile: [\\10.160.50.8\print$\W32X86\2\cns3g.hlp]

  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\AUCPLMNT.DLL]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\aussdrv.dll]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\cnspdc.dll]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\aussapi.dat]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\cns3407.dll]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\CnS3G.cnt]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\NBAPI.DLL]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\NBIPC.DLL]
  Dependntfiles: [\\10.160.50.8\print$\W32X86\2\cns3gum.dll]

  Monitorname: [CPCA Language Monitor2]
  Defaultdatatype: []

```

If we write the “*version 2*” files and the “*version 3*” files into different text files and compare the result, we see this picture:

```

root# sdiff 2-files 3-files

cns3g.dll                cns3g.dll
iR8500sg.xpd             iR8500sg.xpd
cns3gui.dll              cns3gui.dll
cns3g.hlp                cns3g.hlp
AUCPLMNT.DLL            | aucplmNT.dll
                        > ucs32p.dll
                        > tnl32.dll

aussdrv.dll              aussdrv.dll
cnspdc.dll               cnspdc.dll
aussapi.dat              aussapi.dat
cns3407.dll              cns3407.dll
CnS3G.cnt                CnS3G.cnt
NBAPI.DLL                NBAPI.DLL
NBIPC.DLL                NBIPC.DLL
cns3gum.dll              | cpcview.exe
                        > cpcdsp1.exe
                        > cpcqm.exe

```

```

> cpcspl.dll
> cfine32.dll
> cpcr407.dll
> Cpcqm407.hlp
> cpcqm407.cnt
> cns3ggr.dll

```

Do not be fooled! Driver files for each version with identical names may be different in their content, as you can see from this size comparison:

```

root# for i in cns3g.hlp cns3gui.dll cns3g.dll; do \
    smbclient //10.160.50.8/print\$ -U 'Administrator%xxxx' \
    -c "cd W32X86/3; dir $i; cd .. ; cd 2; dir $i"; \
done

CNS3G.HLP          A    122981  Thu May 30 02:31:00 2002
CNS3G.HLP          A     99948  Thu May 30 02:31:00 2002

CNS3GUI.DLL        A   1805824  Thu May 30 02:31:00 2002
CNS3GUI.DLL        A   1785344  Thu May 30 02:31:00 2002

CNS3G.DLL          A   1145088  Thu May 30 02:31:00 2002
CNS3G.DLL          A     15872  Thu May 30 02:31:00 2002

```

In my example were even more differences than shown here. Conclusion: you must be careful to select the correct driver files for each driver version. Don't rely on the names alone and don't interchange files belonging to different driver versions.

17.8.6 Samba and Printer Ports

Windows NT/2000 print servers associate a port with each printer. These normally take the form of LPT1:, COM1:, FILE:, and so on. Samba must also support the concept of ports associated with a printer. By default, only one printer port, named “*Samba Printer Port*”, exists on a system. Samba does not really need such a “*port*” in order to print; rather it is a requirement of Windows clients. They insist on being told about an available port when they request this information, otherwise they throw an error message at you. So Samba fakes the port information to keep the Windows clients happy.

Samba does not support the concept of **Printer Pooling** internally either. Printer Pooling assigns a logical printer to multiple ports as a form of load balancing or fail over.

If you require multiple ports be defined for some reason or another (my users and my boss should not know that they are working with Samba), configure *enumports command* which can be used to define an external program that generates a listing of ports on a system.

17.8.7 Avoiding Common Client Driver Misconfiguration

So now the printing works, but there are still problems. Most jobs print well, some do not print at all. Some jobs have problems with fonts, which do not look good. Some jobs print fast and some are dead-slow. We cannot cover it all, but we want to encourage you to read the brief paragraph about “*Avoiding the Wrong PostScript Driver Settings*” in the CUPS Printing part of this document.

17.9 The Imprints Toolset

The Imprints tool set provides a UNIX equivalent of the Windows NT Add Printer Wizard. For complete information, please refer to the Imprints Web site at <http://imprints.sourceforge.net/> as well as the documentation included with the imprints source distribution. This section only provides a brief introduction to the features of Imprints.

Unfortunately, the Imprints toolset is no longer maintained. As of December 2000, the project is in need of a new maintainer. The most important skill to have is Perl coding and an interest in MS-RPC-based printing used in Samba. If you wish to volunteer, please coordinate your efforts on the Samba technical mailing list. The toolset is still in usable form, but only for a series of older printer models where there are prepared packages to use. Packages for more up-to-date print devices are needed if Imprints should have a future.

17.9.1 What is Imprints?

Imprints is a collection of tools for supporting these goals:

- Providing a central repository of information regarding Windows NT and 95/98 printer driver packages.
- Providing the tools necessary for creating the Imprints printer driver packages.
- Providing an installation client that will obtain printer drivers from a central Internet (or intranet) Imprints Server repository and install them on remote Samba and Windows NT4 print servers.

17.9.2 Creating Printer Driver Packages

The process of creating printer driver packages is beyond the scope of this document (refer to `Imprints.txt` also included with the Samba distribution for more information). In short, an Imprints driver package is a gzipped tarball containing the driver files, related INF files, and a control file needed by the installation client.

17.9.3 The Imprints Server

The Imprints server is really a database server that may be queried via standard HTTP mechanisms. Each printer entry in the database has an associated URL for the actual downloading of the package. Each package is digitally signed via GnuPG which can be used

to verify that the package downloaded is actually the one referred in the Imprints database. It is strongly recommended that this security check not be disabled.

17.9.4 The Installation Client

More information regarding the Imprints installation client is available from the the documentation file `Imprints-Client-HOWTO.ps` that is included with the Imprints source package. The Imprints installation client comes in two forms:

- A set of command line Perl scripts.
- A GTK+ based graphical interface to the command line Perl scripts.

The installation client (in both forms) provides a means of querying the Imprints database server for a matching list of known printer model names as well as a means to download and install the drivers on remote Samba and Windows NT print servers.

The basic installation process is in four steps and Perl code is wrapped around `smbclient` and `rpcclient`.

- For each supported architecture for a given driver:
 1. `rpcclient`: Get the appropriate upload directory on the remote server.
 2. `smbclient`: Upload the driver files.
 3. `rpcclient`: Issues an `AddPrinterDriver()` MS-RPC.
- `rpcclient`: Issue an `AddPrinterEx()` MS-RPC to actually create the printer.

One of the problems encountered when implementing the Imprints tool set was the name space issues between various supported client architectures. For example, Windows NT includes a driver named “*Apple LaserWriter II NTX v51.8*” and Windows 95 calls its version of this driver “*Apple LaserWriter II NTX*”.

The problem is how to know what client drivers have been uploaded for a printer. An astute reader will remember that the Windows NT Printer Properties dialog only includes space for one printer driver name. A quick look in the Windows NT 4.0 system registry at:

```
HKLM\System\CurrentControlSet\Control\Print\Environment
```

will reveal that Windows NT always uses the NT driver name. This is okay as Windows NT always requires that at least the Windows NT version of the printer driver is present. Samba does not have the requirement internally, therefore, “*How can you use the NT driver name if it has not already been installed?*”

The way of sidestepping this limitation is to require that all Imprints printer driver packages include both the Intel Windows NT and 95/98 printer drivers and that the NT driver is installed first.

17.10 Adding Network Printers without User Interaction

The following MS Knowledge Base article may be of some help if you need to handle Windows 2000 clients: *How to Add Printers with No User Interaction in Windows 2000*,

(<http://support.microsoft.com/default.aspx?scid=kb;en-us;189105>³). It also applies to Windows XP Professional clients. The ideas sketched out in this section are inspired by this article, which describes a commandline method that can be applied to install network and local printers and their drivers. This is most useful if integrated in Logon Scripts. You can see what options are available by typing in the command prompt (**DOS box**):

```
rundll32 printui.dll,PrintUIEntry /?
```

A window pops up that shows you all of the commandline switches available. An extensive list of examples is also provided. This is only for Win 200x/XP, it does not work on Windows NT. Windows NT probably has some other tools in the respective Resource Kit. Here is a suggestion about what a client logon script might contain, with a short explanation of what the lines actually do (it works if 200x/XP Windows clients access printers via Samba, and works for Windows-based print servers too):

```
rundll32 printui.dll,PrintUIEntry /dn /n "\\cupsserver\infotec2105-IPDS" /q
rundll32 printui.dll,PrintUIEntry /in /n "\\cupsserver\infotec2105-PS"
rundll32 printui.dll,PrintUIEntry /y /n "\\cupsserver\infotec2105-PS"
```

Here is a list of the used commandline parameters:

/dn — deletes a network printer

/q — quiet modus

/n — names a printer

/in — adds a network printer connection

/y — sets printer as default printer

- Line 1 deletes a possibly existing previous network printer *infotec2105-IPDS* (which had used native Windows drivers with LPRng that were removed from the server that was converted to CUPS). The **/q** at the end eliminates Confirm or error dialog boxes from popping up. They should not be presented to the user logging on.
- Line 2 adds the new printer *infotec2105-PS* (which actually is the same physical device but is now run by the new CUPS printing system and associated with the CUPS/Adobe PS drivers). The printer and its driver must have been added to Samba prior to the user logging in (e.g., by a procedure as discussed earlier in this chapter, or by running **cupsaddsmb**). The driver is now auto-downloaded to the client PC where the user is about to log in.
- Line 3 sets the default printer to this new network printer (there might be several other printers installed with this same method and some may be local as well, so we

³<http://support.microsoft.com/default.aspx?scid=kb;en-us;189105>

decide for a default printer). The default printer selection may, of course, be different for different users.

The second line only works if the printer *infotec2105-PS* has an already working print queue on the `cupserver`, and if the printer drivers have been successfully uploaded (via the **APW**, **smbclient/rpcclient**, or **cupsaddsmb**) into the `[print$]` driver repository of Samba. Some Samba versions prior to version 3.0 required a re-start of `smbd` after the printer install and the driver upload, otherwise the script (or any other client driver download) would fail.

Since there no easy way to test for the existence of an installed network printer from the logon script, do not bother checking, just allow the deinstallation/reinstallation to occur every time a user logs in; it's really quick anyway (1 to 2 seconds).

The additional benefits for this are:

- It puts in place any printer default setup changes automatically at every user logon.
- It allows for “*roaming*” users’ login into the domain from different workstations.

Since network printers are installed per user, this much simplifies the process of keeping the installation up-to-date. The few extra seconds at logon time will not really be noticeable. Printers can be centrally added, changed and deleted at will on the server with no user intervention required from the clients (you just need to keep the logon scripts up-to-date).

17.11 The `addprinter` Command

The **addprinter** command can be configured to be a shell script or program executed by Samba. It is triggered by running the APW from a client against the Samba print server. The APW asks the user to fill in several fields (such as printer name, driver to be used, comment, port monitor, and so on). These parameters are passed on to Samba by the APW. If the `addprinter` command is designed in a way that it can create a new printer (through writing correct `printcap` entries on legacy systems, or execute the **lpadmin** command on more modern systems) and create the associated share in, then the APW will in effect really create a new printer on Samba and the UNIX print subsystem!

17.12 Migration of Classical Printing to Samba

The basic NT-style printer driver management has not changed considerably in 3.0 over the 2.2.x releases (apart from many small improvements). Here migration should be quite easy, especially if you followed previous advice to stop using deprecated parameters in your setup. For migrations from an existing 2.0.x setup, or if you continued Windows 9x/Me-style printing in your Samba 2.2 installations, it is more of an effort. Please read the appropriate release notes and the HOWTO Collection for Samba-2.2.x. You can follow several paths. Here are possible scenarios for migration:

- You need to study and apply the new Windows NT printer and driver support. Previously used parameters *printer driver file*, *printer driver* and *printer driver location* are no longer supported.

- If you want to take advantage of Windows NT printer driver support, you also need to migrate the Windows 9x/Me drivers to the new setup.
- An existing `printers.def` file (the one specified in the now removed parameter *printer driver file*) will no longer work with Samba-3. In 3.0, `smbd` attempts to locate a Windows 9x/Me driver files for the printer in `[print$]` and additional settings in the TDB and only there; if it fails, it will *not* (as 2.2.x used to do) drop down to using a `printers.def` (and all associated parameters). The `make_printerdef` tool is removed and there is no backward compatibility for this.
- You need to install a Windows 9x/Me driver into the `[print$]` share for a printer on your Samba host. The driver files will be stored in the “`WIN40/0`” subdirectory of `[print$]`, and some other settings and information go into the printing-related TDBs.
- If you want to migrate an existing `printers.def` file into the new setup, the only current solution is to use the Windows NT APW to install the NT drivers and the 9x/Me drivers. This can be scripted using `smbclient` and `rpcclient`. See the Imprints installation client at:

`http://imprints.sourceforge.net/`

for an example. See also the discussion of `rpcclient` usage in the “*CUPS Printing*” section.

17.13 Publishing Printer Information in Active Directory or LDAP

This will be addressed in a later update of this document. If you wish to volunteer your services to help document this, please contact John H Terpstra.⁴

17.14 Common Errors

17.14.1 I Give My Root Password but I Do Not Get Access

Do not confuse the root password which is valid for the UNIX system (and in most cases stored in the form of a one-way hash in a file named `/etc/shadow`), with the password used to authenticate against Samba. Samba does not know the UNIX password. Root access to Samba resources requires that a Samba account for root must first be created. This is done with the `smbpasswd` command as follows:

```
root# smbpasswd -a root
New SMB password: secret
Retype new SMB password: secret
```

⁴[mail://jht@samba.org](mailto://jht@samba.org)

17.14.2 My Print Jobs Get Spooled into the Spooling Directory, but Then Get Lost

Do not use the existing UNIX print system spool directory for the Samba spool directory. It may seem convenient and a savings of space, but it only leads to problems. The two must be separate.

CUPS PRINTING SUPPORT

18.1 Introduction

18.1.1 Features and Benefits

The Common UNIX Print System (CUPS¹) has become quite popular. All major Linux distributions now ship it as their default printing system. To many, it is still a mystical tool. Mostly, it just works. People tend to regard it as a “*black box*” that they do not want to look into as long as it works. But once there is a little problem, they are in trouble to find out where to start debugging it. Refer to the chapter “*Classical Printing*” that contains a lot of information that is relevant for CUPS.

CUPS sports quite a few unique and powerful features. While their basic functions may be grasped quite easily, they are also new. Because they are different from other, more traditional printing systems, it is best not to try and apply any prior knowledge about printing to this new system. Rather, try to understand CUPS from the beginning. This documentation will lead you to a complete understanding of CUPS. Let’s start with the most basic things first.

18.1.2 Overview

CUPS is more than just a print spooling system. It is a complete printer management system that complies with the new Internet Printing Protocol (IPP). IPP is an industry and Internet Engineering Task Force (IETF) standard for network printing. Many of its functions can be managed remotely (or locally) via a Web browser (giving you a platform-independent access to the CUPS print server). Additionally, it has the traditional command line and several more modern GUI interfaces (GUI interfaces developed by third parties, like KDE’s overwhelming KDEPrint²).

CUPS allows creation of “*raw*” printers (i.e., no print file format translation) as well as “*smart*” printers (i.e., CUPS does file format conversion as required for the printer). In many ways this gives CUPS similar capabilities to the MS Windows print monitoring system. Of course, if you are a CUPS advocate, you would argue that CUPS is better! In any case, let

¹<http://www.cups.org/>

²<http://printing.kde.org/>

us now move on to explore how one may configure CUPS for interfacing with MS Windows print clients via Samba.

18.2 Basic CUPS Support Configuration

Printing with CUPS in the most basic `smb.conf` setup in Samba-3.0 (as was true for 2.2.x) only needs two settings: `printing = cups` and `printcap = cups`. CUPS does not need a printcap file. However, the `cupsd.conf` configuration file knows of two related directives that control how such a file will be automatically created and maintained by CUPS for the convenience of third-party applications (example: `Printcap /etc/printcap` and `PrintcapFormat BSD`). Legacy programs often require the existence of a printcap file containing printer names or they will refuse to print. Make sure CUPS is set to generate and maintain a printcap file. For details, see **man cupsd.conf** and other CUPS-related documentation, like the wealth of documents on your CUPS server itself: <http://localhost:631/documentation.html>.

18.2.1 Linking `smbd` with `libcups.so`

Samba has a special relationship to CUPS. Samba can be compiled with CUPS library support. Most recent installations have this support enabled. Per default, CUPS linking is compiled into `smbd` and other Samba binaries. Of course, you can use CUPS even if Samba is not linked against `libcups.so` — but there are some differences in required or supported configuration.

When Samba is compiled against `libcups`, `printcap = cups` uses the CUPS API to list printers, submit jobs, query queues, and so on. Otherwise it maps to the System V commands with an additional `-oraw` option for printing. On a Linux system, you can use the **ldd** utility to find out details (`ldd` may not be present on other OS platforms, or its function may be embodied by a different command):

```
root# ldd 'which smbd'
libssl.so.0.9.6 => /usr/lib/libssl.so.0.9.6 (0x4002d000)
libcrypto.so.0.9.6 => /usr/lib/libcrypto.so.0.9.6 (0x4005a000)
libcups.so.2 => /usr/lib/libcups.so.2 (0x40123000)
[...]
```

The line `libcups.so.2 => /usr/lib/libcups.so.2 (0x40123000)` shows there is CUPS support compiled into this version of Samba. If this is the case, and `printing = cups` is set, then *any otherwise manually set print command in `smb.conf` is ignored*. This is an important point to remember!

TIP



Should it be necessary, for any reason, to set your own print commands, you can do this by setting *printing = sysv*. However, you will lose all the benefits of tight CUPS/Samba integration. When you do this you must manually configure the printing system commands (most important: *print command*; other commands are *lppause command*, *lpresume command*, *lpq command*, *lprm command*, *queuepause command* and *queue resume command*).

18.2.2 Simple smb.conf Settings for CUPS

To summarize, Example 18.1 shows simplest printing-related setup for `smb.conf` to enable basic CUPS support:

Example 18.1. Simplest printing-related `smb.conf`

```
[global]
load printers = yes
printing = cups
printcap name = cups

[printers]
comment = All Printers
path = /var/spool/samba
browseable = no
public = yes
guest ok = yes
writable = no
printable = yes
printer admin = root, @ntadmins
```

This is all you need for basic printing setup for CUPS. It will print all graphic, text, PDF, and PostScript files submitted from Windows clients. However, most of your Windows users would not know how to send these kinds of files to print without opening a GUI application. Windows clients tend to have local printer drivers installed, and the GUI application's print buttons start a printer driver. Your users also rarely send files from the command line. Unlike UNIX clients, they hardly submit graphic, text or PDF formatted files directly to the spooler. They nearly exclusively print from GUI applications with a “*printer driver*” hooked in between the application's native format and the print-data-stream. If the backend printer is not a PostScript device, the print data stream is “*binary*,” sensible only for the target printer. Read on to learn which problem this may cause and how to avoid it.

18.2.3 More Complex CUPS smb.conf Settings

Example 18.2 is a slightly more complex printing-related setup for `smb.conf`. It enables general CUPS printing support for all printers, but defines one printer share, which is set up differently.

Example 18.2. Overriding global CUPS settings for one printer

```
[global]
printing = cups
printcap name = cups
load printers = yes

[printers]
comment = All Printers
path = /var/spool/samba
public = yes
guest ok = yes
writable = no
printable = yes
printer admin = root, @ntadmins

[special_printer]
comment = A special printer with his own settings
path = /var/spool/samba-special
printing = sysv
printcap = lpstat
print command = echo "NEW: 'date': printfile %f" >> /tmp/smbprn.log ; \
echo " 'date': p-%p s-%s f-%f" >> /tmp/smbprn.log ; \
echo " 'date': j-%j J-%J z-%z c-%c" >> /tmp/smbprn.log : rm %f
public = no
guest ok = no
writeable = no
printable = yes
printer admin = kurt
hosts deny = 0.0.0.0
hosts allow = turbo_xp, 10.160.50.23, 10.160.51.60
```

This special share is only there for testing purposes. It does not write the print job to a file. It just logs the job parameters known to Samba into the `/tmp/smbprn.log` file and deletes the jobfile. Moreover, the `printer admin` of this share is “kurt” (not the “@ntadmins” group), guest access is not allowed, the share isn’t published to the Network Neighborhood (so you need to know it is there), and it only allows access from only three hosts. To prevent CUPS kicking in and taking over the print jobs for that share, we need to set `printing = sysv` and `printcap = lpstat`.

18.3 Advanced Configuration

Before we delve into all the configuration options, let us clarify a few points. *Network printing needs to be organized and setup correctly.* This frequently doesn't happen. Legacy systems or small business LAN environments often lack design and good housekeeping.

18.3.1 Central Spooling vs. “Peer-to-Peer” Printing

Many small office or home networks, as well as badly organized larger environments, allow each client a direct access to available network printers. This is generally a bad idea. It often blocks one client's access to the printer when another client's job is printing. It might freeze the first client's application while it is waiting to get rid of the job. Also, there are frequent complaints about various jobs being printed with their pages mixed with each other. A better concept is the usage of a print server: it routes all jobs through one central system, which responds immediately, takes jobs from multiple concurrent clients at the same time, and in turn transfers them to the printer(s) in the correct order.

18.3.2 Raw Print Serving — Vendor Drivers on Windows Clients

Most traditionally configured UNIX print servers acting on behalf of Samba's Windows clients represented a really simple setup. Their only task was to manage the “raw” spooling of all jobs handed to them by Samba. This approach meant that the Windows clients were expected to prepare the print job file that its ready to be sent to the printing device. Here is a native (vendor-supplied) Windows printer driver for the target device needed to be installed on each and every client.

It is possible to configure CUPS, Samba and your Windows clients in the same traditional and simple way. When CUPS printers are configured for RAW print-through mode operation, it is the responsibility of the Samba client to fully render the print job (file). The file must be sent in a format that is suitable for direct delivery to the printer. Clients need to run the vendor-provided drivers to do this. In this case, CUPS will not do any print file format conversion work.

18.3.3 Installation of Windows Client Drivers

The printer drivers on the Windows clients may be installed in two functionally different ways:

- Manually install the drivers locally on each client, one by one; this yields the old *LanMan* style printing and uses a `\\sambaserver\printershare` type of connection.
- Deposit and prepare the drivers (for later download) on the print server (Samba); this enables the clients to use “*Point'n'Print*” to get drivers semi-automatically installed the first time they access the printer; with this method NT/200x/XP clients use the *SPOOLSS/MS-RPC* type printing calls.

The second method is recommended for use over the first.

18.3.4 Explicitly Enable “raw” Printing for *application/octet-stream*

If you use the first option (drivers are installed on the client side), there is one setting to take care of: CUPS needs to be told that it should allow “raw” printing of deliberate (binary) file formats. The CUPS files that need to be correctly set for RAW mode printers to work are:

- `/etc/cups/mime.types`
- `/etc/cups/mime.convs`

Both contain entries (at the end of the respective files) which must be uncommented to allow RAW mode operation. In `/etc/cups/mime.types`, make sure this line is present:

```
application/octet-stream
```

In `/etc/cups/mime.convs`, have this line:

```
application/octet-stream application/vnd.cups-raw 0 -
```

If these two files are not set up correctly for raw Windows client printing, you may encounter the dreaded `Unable to convert file 0` in your CUPS `error.log` file.

NOTE



Editing the `mime.convs` and the `mime.types` file does not *enforce* “raw” printing, it only *allows* it.

CUPS being a more security-aware printing system than traditional ones does not by default allow a user to send deliberate (possibly binary) data to printing devices. This could be easily abused to launch a “*Denial of Service*” attack on your printer(s), causing at least the loss of a lot of paper and ink. “*Unknown*” data are tagged by CUPS as *MIME type: application/octet-stream* and not allowed to go to the printer. By default, you can only send other (known) MIME types “raw”. Sending data “raw” means that CUPS does not try to convert them and passes them to the printer untouched (see the next chapter for even more background explanations).

This is all you need to know to get the CUPS/Samba combo printing “raw” files prepared by Windows clients, which have vendor drivers locally installed. If you are not interested in background information about more advanced CUPS/Samba printing, simply skip the remaining sections of this chapter.

18.3.5 Driver Upload Methods

This section describes three familiar methods, plus one new one, by which printer drivers may be uploaded.

If you want to use the MS-RPC type printing, you must upload the drivers onto the Samba server first (*[print\$]* share). For a discussion on how to deposit printer drivers on the Samba host (so the Windows clients can download and use them via “*Point’n’Print*”), please refer to the previous chapter of this HOWTO Collection. There you will find a description or reference to three methods of preparing the client drivers on the Samba server:

- The GUI, “*Add Printer Wizard*” *upload-from-a-Windows-client* method.
- The command line, “*smbclient/rpcclient*” *upload-from-a-UNIX-workstation* method.
- The Imprints Toolset method.

These three methods apply to CUPS all the same. A new and more convenient way to load the Windows drivers into Samba is provided if you use CUPS:

- the *cupsaddsmb* utility.

cupsaddsmb is discussed in much detail further below. But we first explore the CUPS filtering system and compare the Windows and UNIX printing architectures.

18.4 Advanced Intelligent Printing with PostScript Driver Download

We now know how to set up a “*dump*” printserver, that is, a server which is spooling printjobs “*raw*”, leaving the print data untouched.

Possibly you need to setup CUPS in a smarter way. The reasons could be manifold:

- Maybe your boss wants to get monthly statistics: Which printer did how many pages? What was the average data size of a job? What was the average print run per day? What are the typical hourly peaks in printing? Which department prints how much?
- Maybe you are asked to setup a print quota system: Users should not be able to print more jobs, once they have surpassed a given limit per period.
- Maybe your previous network printing setup is a mess and must be re-organized from a clean beginning.
- Maybe you have experiencing too many “*blue screens*” originating from poorly debugged printer drivers running in NT “*kernel mode*”?

These goals cannot be achieved by a raw print server. To build a server meeting these requirements, you’ll first need to learn about how CUPS works and how you can enable its features.

What follows is the comparison of some fundamental concepts for Windows and UNIX printing; then follows a description of the CUPS filtering system, how it works and how you can tweak it.

18.4.1 GDI on Windows – PostScript on UNIX

Network printing is one of the most complicated and error-prone day-to-day tasks any user or administrator may encounter. This is true for all OS platforms. And there are reasons for this.

You can't expect most file formats to just throw them toward printers and they get printed. There needs to be a file format conversion in between. The problem is that there is no common standard for print file formats across all manufacturers and printer types. While PostScript (trademark held by Adobe) and, to an extent, PCL (trademark held by HP) have developed into semi-official “*standards*” by being the most widely used PDLs Page Description Languages (PDLs), there are still many manufacturers who “*roll their own*” (their reasons may be unacceptable license fees for using printer-embedded PostScript interpreters, and so on).

18.4.2 Windows Drivers, GDI and EMF

In Windows OS, the format conversion job is done by the printer drivers. On MS Windows OS platforms all application programmers have at their disposal a built-in API, the Graphical Device Interface (GDI), as part and parcel of the OS itself to base themselves on. This GDI core is used as one common unified ground for all Windows programs to draw pictures, fonts and documents *on screen* as well as *on paper* (print). Therefore, printer driver developers can standardize on a well-defined GDI output for their own driver input. Achieving WYSIWYG (“*What You See Is What You Get*”) is relatively easy, because the on-screen graphic primitives, as well as the on-paper drawn objects, come from one common source. This source, the GDI, often produces a file format called Enhanced MetaFile (EMF). The EMF is processed by the printer driver and converted to the printer-specific file format.

NOTE



To the GDI foundation in MS Windows, Apple has chosen to put paper and screen output on a common foundation for their (BSD-UNIX-based, did you know?) Mac OS X and Darwin Operating Systems. Their *Core Graphic Engine* uses a *PDF* derivative for all display work.

18.4.3 UNIX Printfile Conversion and GUI Basics

In UNIX and Linux, there is no comparable layer built into the OS kernel(s) or the X (screen display) server. Every application is responsible for itself to create its print output. Fortunately, most use PostScript and that at least gives some common ground. Unfortunately, there are many different levels of quality for this PostScript. And worse, there is a huge difference (and no common root) in the way the same document is displayed on screen and how it is presented on paper. WYSIWYG is more difficult to achieve. This goes back to the time, decades ago, when the predecessors of X.org, designing the UNIX foundations and protocols for Graphical User Interfaces, refused to take responsibility for “*paper output*”

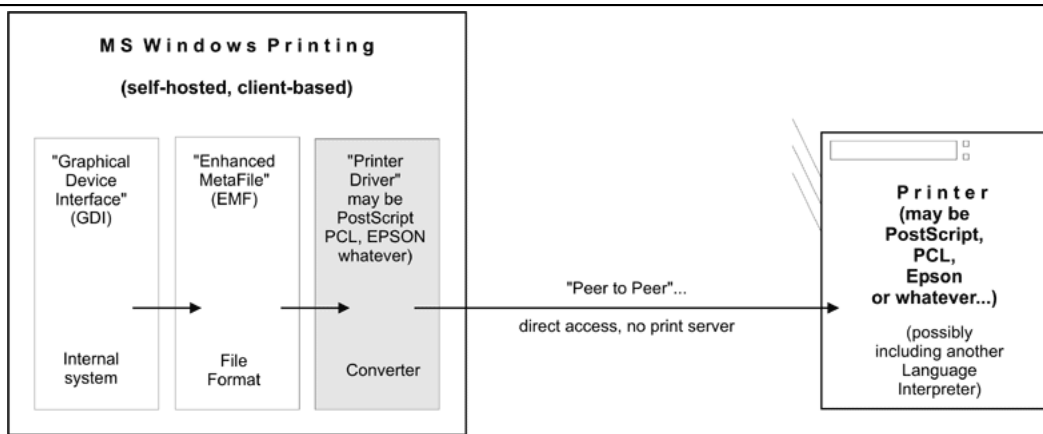


Figure 18.1. Windows printing to a local printer.

also, as some had demanded at the time, and restricted itself to “*on-screen only*.” (For some years now, the “*Xprint*” project has been under development, attempting to build printing support into the X framework, including a PostScript and a PCL driver, but it is not yet ready for prime time.) You can see this unfavorable inheritance up to the present day by looking into the various “*font*” directories on your system; there are separate ones for fonts used for X display and fonts to be used on paper.

The PostScript programming language is an “*invention*” by Adobe Inc., but its specifications have been published to the full. Its strength lies in its powerful abilities to describe graphical objects (fonts, shapes, patterns, lines, curves, and dots), their attributes (color, linewidth) and the way to manipulate (scale, distort, rotate, shift) them. Because of its open specification, anybody with the skill can start writing his own implementation of a PostScript interpreter and use it to display PostScript files on screen or on paper. Most graphical output devices are based on the concept of “*raster images*” or “*pixels*” (one notable exception is pen plotters). Of course, you can look at a PostScript file in its textual form and you will be reading its PostScript code, the language instructions which need to be interpreted by a rasterizer. Rasterizers produce pixel images, which may be displayed on screen by a viewer program or on paper by a printer.

18.4.4 PostScript and Ghostscript

So, UNIX is lacking a common ground for printing on paper and displaying on screen. Despite this unfavorable legacy for UNIX, basic printing is fairly easy if you have PostScript printers at your disposal. The reason is these devices have a built-in PostScript language “*interpreter*,” also called a Raster Image Processor (RIP) (which makes them more expensive than other types of printers); throw PostScript toward them, and they will spit out your printed pages. Their RIP is doing all the hard work of converting the PostScript drawing commands into a bitmap picture as you see it on paper, in a resolution as done by your printer. This is no different to PostScript printing a file from a Windows origin.

NOTE



Traditional UNIX programs and printing systems — while using PostScript — are largely not PPD-aware. PPDs are “*PostScript Printer Description*” files. They enable you to specify and control all options a printer supports: duplexing, stapling and punching. Therefore, UNIX users for a long time couldn’t choose many of the supported device and job options, unlike Windows or Apple users. But now there is CUPS.

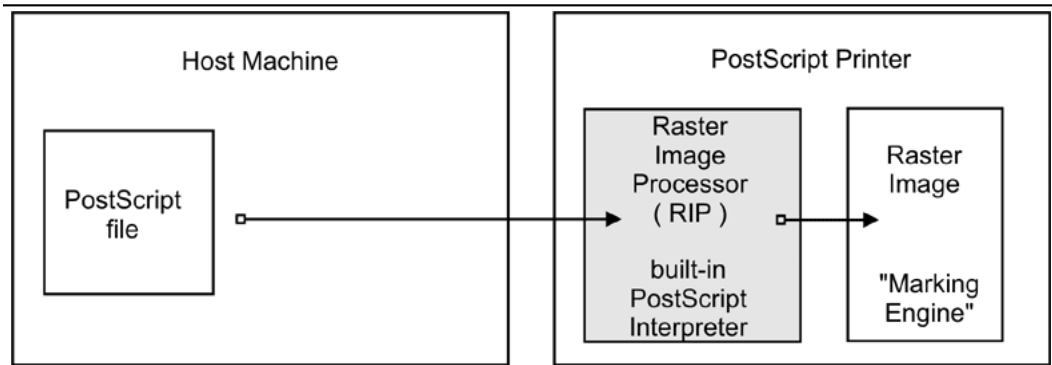


Figure 18.2. Printing to a PostScript printer.

However, there are other types of printers out there. These do not know how to print PostScript. They use their own Page Description Language (PDL, often proprietary). To print to them is much more demanding. Since your UNIX applications mostly produce PostScript, and since these devices do not understand PostScript, you need to convert the printfiles to a format suitable for your printer on the host before you can send it away.

18.4.5 Ghostscript — the Software RIP for Non-PostScript Printers

Here is where Ghostscript kicks in. Ghostscript is the traditional (and quite powerful) PostScript interpreter used on UNIX platforms. It is a RIP in software, capable of doing a *lot* of file format conversions for a very broad spectrum of hardware devices as well as software file formats. Ghostscript technology and drivers are what enable PostScript printing to non-PostScript hardware.

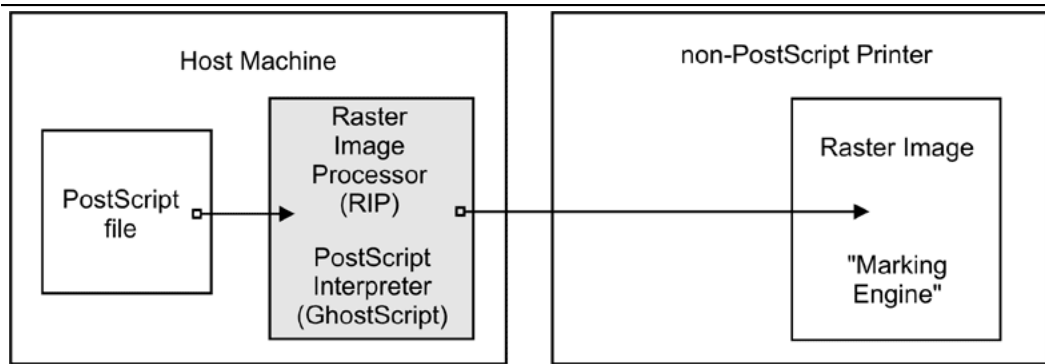


Figure 18.3. Ghostscript as a RIP for non-postscript printers.

TIP



Use the `gs -h` command to check for all built-in “*devices*” of your Ghostscript version. If you specify a parameter of `-sDEVICE=png256` on your Ghostscript command line, you are asking Ghostscript to convert the input into a PNG file. Naming a “*device*” on the command line is the most important single parameter to tell Ghostscript exactly how it should render the input. New Ghostscript versions are released at fairly regular intervals, now by artofcode LLC. They are initially put under the “*AFPL*” license, but re-released under the GNU GPL as soon as the next AFPL version appears. GNU Ghostscript is probably the version installed on most Samba systems. But it has some deficiencies. Therefore, ESP Ghostscript was developed as an enhancement over GNU Ghostscript, with lots of bug-fixes, additional devices and improvements. It is jointly maintained by developers from CUPS, Gimp-Print, MandrakeSoft, SuSE, RedHat, and Debian. It includes the “*cups*” device (essential to print to non-PS printers from CUPS).

18.4.6 PostScript Printer Description (PPD) Specification

While PostScript in essence is a Page Description Language (PDL) to represent the page layout in a device-independent way, real-world print jobs are always ending up being output on hardware with device-specific features. To take care of all the differences in hardware and to allow for innovations, Adobe has specified a syntax and file format for PostScript Printer Description (PPD) files. Every PostScript printer ships with one of these files.

PPDs contain all the information about general and special features of the given printer model: Which different resolutions can it handle? Does it have a Duplexing Unit? How many paper trays are there? What media types and sizes does it take? For each item, it also names the special command string to be sent to the printer (mostly inside the PostScript file) in order to enable it.

Information from these PPDs is meant to be taken into account by the printer drivers. Therefore, installed as part of the Windows PostScript driver for a given printer is the printer's PPD. Where it makes sense, the PPD features are presented in the drivers' UI dialogs to display to the user a choice of print options. In the end, the user selections are somehow written (in the form of special PostScript, PJJ, JCL or vendor-dependent commands) into the PostScript file created by the driver.

WARNING



A PostScript file that was created to contain device-specific commands for achieving a certain print job output (e.g., duplexed, stapled and punched) on a specific target machine, may not print as expected, or may not be printable at all on other models; it also may not be fit for further processing by software (e.g., by a PDF distilling program).

18.4.7 Using Windows-Formatted Vendor PPDs

CUPS can handle all spec-compliant PPDs as supplied by the manufacturers for their PostScript models. Even if a vendor might not have mentioned our favorite OS in his manuals and brochures, you can safely trust this: *If you get the Windows NT version of the PPD, you can use it unchanged in CUPS* and thus access the full power of your printer just like a Windows NT user could!

TIP



To check the spec compliance of any PPD online, go to <http://www.cups.org/testppd.php> and upload your PPD. You will see the results displayed immediately. CUPS in all versions after 1.1.19 has a much more strict internal PPD parsing and checking code enabled; in case of printing trouble, this online resource should be one of your first pitstops.

WARNING



For real PostScript printers, *do not* use the *Foomatic* or *cupsomatic* PPDs from [Linuxprinting.org](http://linuxprinting.org). With these devices, the original vendor-provided PPDs are always the first choice!

TIP

If you are looking for an original vendor-provided PPD of a specific device, and you know that an NT4 box (or any other Windows box) on your LAN has the PostScript driver installed, just use **smbclient //NT4-box/print/\$ -U username** to access the Windows directory where all printer driver files are stored. First look in the W32X86/2 subdir for the PPD you are seeking.

18.4.8 CUPS Also Uses PPDs for Non-PostScript Printers

CUPS also uses specially crafted PPDs to handle non-PostScript printers. These PPDs are usually not available from the vendors (and no, you can't just take the PPD of a PostScript printer with the same model name and hope it works for the non-PostScript version too). To understand how these PPDs work for non-PS printers, we first need to dive deeply into the CUPS filtering and file format conversion architecture. Stay tuned.

18.5 The CUPS Filtering Architecture

The core of the CUPS filtering system is based on Ghostscript. In addition to Ghostscript, CUPS uses some other filters of its own. You (or your OS vendor) may have plugged in even more filters. CUPS handles all data file formats under the label of various MIME types. Every incoming printfile is subjected to an initial auto-typing. The auto-typing determines its given MIME type. A given MIME type implies zero or more possible filtering chains relevant to the selected target printer. This section discusses how MIME types recognition and conversion rules interact. They are used by CUPS to automatically setup a working filtering chain for any given input data format.

If CUPS rasterizes a PostScript file natively to a bitmap, this is done in two stages:

- The first stage uses a Ghostscript device named “*cups*” (this is since version 1.1.15) and produces a generic raster format called “*CUPS raster*”.
- The second stage uses a “*raster driver*” that converts the generic CUPS raster to a device-specific raster.

Make sure your Ghostscript version has the “*cups*” device compiled in (check with **gs -h | grep cups**). Otherwise you may encounter the dreaded **Unable to convert file 0** in your CUPS error.log file. To have “*cups*” as a device in your Ghostscript, you either need to patch GNU Ghostscript and re-compile, or use ESP Ghostscript³. The superior alternative is ESP Ghostscript. It supports not just CUPS, but 300 other devices too (while GNU Ghostscript supports only about 180). Because of this broad output device support, ESP Ghostscript is the first choice for non-CUPS spoolers, too. It is now recommended by Linuxprinting.org for all spoolers.

³<http://www.cups.org/ghostscript.php>

CUPS printers may be setup to use external rendering paths. One of the most common is provided by the Foomatic/cupsomatic concept from [Linuxprinting.org](http://www.linuxprinting.org).⁴ This uses the classical Ghostscript approach, doing everything in one step. It does not use the “*cups*” device, but one of the many others. However, even for Foomatic/cupsomatic usage, best results and broadest printer model support is provided by ESP Ghostscript (more about cupsomatic/Foomatic, particularly the new version called now *foomatic-rip*, follows below).

18.5.1 MIME Types and CUPS Filters

CUPS reads the file `/etc/cups/mime.types` (and all other files carrying a `*.types` suffix in the same directory) upon startup. These files contain the MIME type recognition rules that are applied when CUPS runs its auto-typing routines. The rule syntax is explained in the man page for `mime.types` and in the comments section of the `mime.types` file itself. A simple rule reads like this:

```
application/pdf          pdf string(0,%PDF)
```

This means if a filename has either a `.pdf` suffix or if the magic string `%PDF` is right at the beginning of the file itself (offset 0 from the start), then it is a PDF file (*application/pdf*). Another rule is this:

```
application/postscript  ai eps ps string(0,%!) string(0,<04>%!)
```

If the filename has one of the suffixes `.ai`, `.eps`, `.ps` or if the file itself starts with one of the strings `%!` or `<04>%!`, it is a generic PostScript file (*application/postscript*).

WARNING



Don't confuse the other `mime.types` files your system might be using with the one in the `/etc/cups/` directory.

⁴<http://www.linuxprinting.org/>

NOTE



There is an important difference between two similar MIME types in CUPS: one is *application/postscript*, the other is *application/vnd.cups-postscript*. While *application/postscript* is meant to be device independent (job options for the file are still outside the PS file content, embedded in command line or environment variables by CUPS), *application/vnd.cups-postscript* may have the job options inserted into the PostScript data itself (where applicable). The transformation of the generic PostScript (*application/postscript*) to the device-specific version (*application/vnd.cups-postscript*) is the responsibility of the CUPS *pstops* filter. *pstops* uses information contained in the PPD to do the transformation.

CUPS can handle ASCII text, HP-GL, PDF, PostScript, DVI, and many image formats (GIF, PNG, TIFF, JPEG, Photo-CD, SUN-Raster, PNM, PBM, SGI-RGB, and more) and their associated MIME types with its filters.

18.5.2 MIME Type Conversion Rules

CUPS reads the file `/etc/cups/mime.convs` (and all other files named with a `*.convs` suffix in the same directory) upon startup. These files contain lines naming an input MIME type, an output MIME type, a format conversion filter that can produce the output from the input type and virtual costs associated with this conversion. One example line reads like this:

```
application/pdf          application/postscript  33  pdftops
```

This means that the *pdftops* filter will take *application/pdf* as input and produce *application/postscript* as output; the virtual cost of this operation is 33 CUPS-\$. The next filter is more expensive, costing 66 CUPS-:

```
application/vnd.hp-HPGL application/postscript  66  hpgltops
```

This is the *hpgltops*, which processes HP-GL plotter files to PostScript.

```
application/octet-stream
```

Here are two more examples:

```
application/x-shell      application/postscript  33  texttops
```

text/plain application/postscript 33 texttops

The last two examples name the *texttops* filter to work on *text/plain* as well as on *application/x-shell*. (Hint: This differentiation is needed for the syntax highlighting feature of *texttops*).

18.5.3 Filtering Overview

There are many more combinations named in `mime.convs`. However, you are not limited to use the ones pre-defined there. You can plug in any filter you like into the CUPS framework. It must meet, or must be made to meet, some minimal requirements. If you find (or write) a cool conversion filter of some kind, make sure it complies to what CUPS needs and put in the right lines in `mime.types` and `mime.convs`, then it will work seamlessly inside CUPS.

18.5.3.1 Filter requirements

The mentioned “*CUPS requirements*” for filters are simple. Take filenames or `stdin` as input and write to `stdout`. They should take these 5 or 6 arguments: *printer job user title copies options [filename]*

Printer — The name of the printer queue (normally this is the name of the filter being run).

job — The numeric job ID for the job being printed.

user — The string from the originating-user-name attribute.

title — The string from the job-name attribute.

copies — The numeric value from the number-copies attribute.

options — The job options.

filename — (Optionally) The print request file (if missing, filters expected data fed through `stdin`). In most cases, it is easy to write a simple wrapper script around existing filters to make them work with CUPS.

18.5.4 Prefilters

As previously stated, PostScript is the central file format to any UNIX-based printing system. From PostScript, CUPS generates raster data to feed non-PostScript printers.

But what happens if you send one of the supported non-PS formats to print? Then CUPS runs “*pre-filters*” on these input formats to generate PostScript first. There are pre-filters to create PS from ASCII text, PDF, DVI, or HP-GL. The outcome of these filters is always of MIME type *application/postscript* (meaning that any device-specific print options are not yet embedded into the PostScript by CUPS, and that the next filter to be called is *pstops*). Another pre-filter is running on all supported image formats, the *imagetops* filter. Its outcome is always of MIME type *application/vnd.cups-postscript* (not *application/postscript*), meaning it has the print options already embedded into the file.

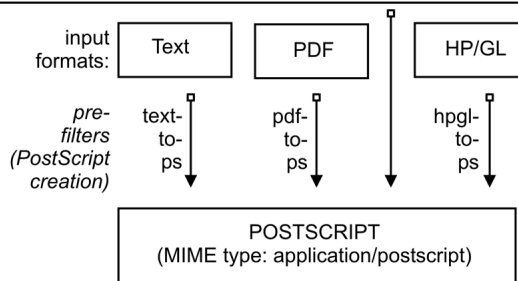


Figure 18.4. Pre-filtering in CUPS to form PostScript.

18.5.5 pstops

pstops is the filter to convert *application/postscript* to *application/vnd.cups-postscript*. It was said above that this filter inserts all device-specific print options (commands to the printer to ask for the duplexing of output, or stapling and punching it, and so on) into the PostScript file.

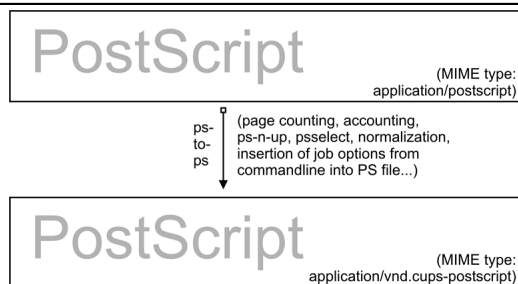


Figure 18.5. Adding device-specific print options.

This is not all. Other tasks performed by it are:

- Selecting the range of pages to be printed (if you choose to print only pages “3, 6, 8-11, 16, 19-21”, or only the odd numbered ones).
- Putting 2 or more logical pages on one sheet of paper (the so-called “*number-up*” function).

- Counting the pages of the job to insert the accounting information into the `/var/log/cups/page_log`.

18.5.6 pstoraster

pstoraster is at the core of the CUPS filtering system. It is responsible for the first stage of the rasterization process. Its input is of MIME type `application/vnd.cups-postscript`; its output is `application/vnd.cups-raster`. This output format is not yet meant to be printable. Its aim is to serve as a general purpose input format for more specialized *raster drivers* that are able to generate device-specific printer data.

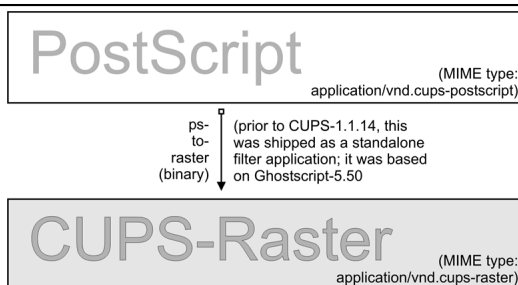


Figure 18.6. PostScript to intermediate raster format.

CUPS raster is a generic raster format with powerful features. It is able to include per-page information, color profiles, and more, to be used by the following downstream raster drivers. Its MIME type is registered with IANA and its specification is, of course, completely open. It is designed to make it quite easy and inexpensive for manufacturers to develop Linux and UNIX raster drivers for their printer models, should they choose to do so. CUPS always takes care for the first stage of rasterization so these vendors do not need to care about Ghostscript complications (in fact, there is currently more than one vendor financing the development of CUPS raster drivers).

CUPS versions before version 1.1.15 were shipping a binary (or source code) standalone filter, named *pstoraster*. *pstoraster* was derived from GNU Ghostscript 5.50, and could be installed besides and in addition to any GNU or AFPL Ghostscript package without conflicting.

>From version 1.1.15, this has changed. The functions for this have been integrated back into Ghostscript (now based on GNU Ghostscript version 7.05). The *pstoraster* filter is now a simple shell script calling `gs` with the `-sDEVICE=cups` parameter. If your Ghostscript does not show a success on asking for `gs -h |grep cups`, you might not be able to print. Update your Ghostscript.

18.5.7 imagetops and imageraster

In the section about pre-filters, we mentioned the pre-filter that generates PostScript from image formats. The *imageraster* filter is used to convert directly from image to raster,

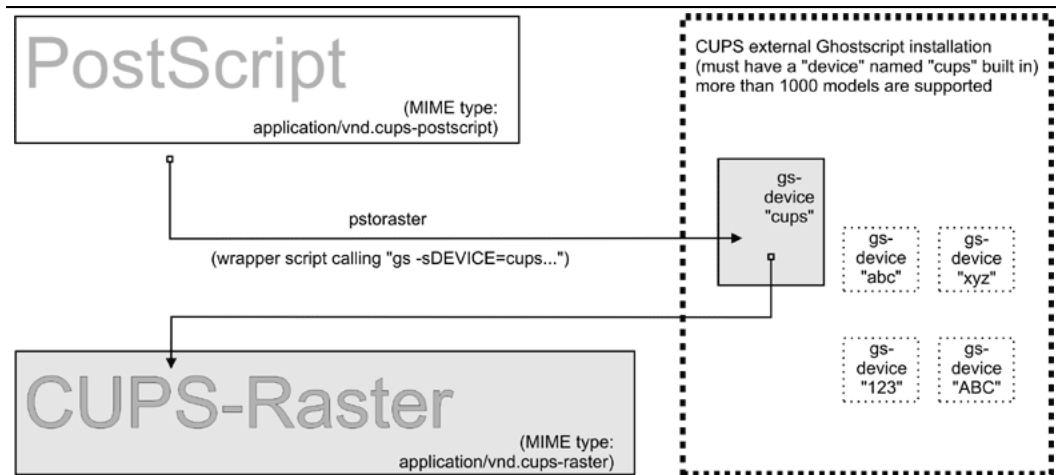


Figure 18.7. CUPS-raster production using Ghostscript.

without the intermediate PostScript stage. It is used more often than the above mentioned pre-filters. A summarizing flowchart of image file filtering is shown in Figure 18.8.

18.5.8 `rasterto` [printers specific]

CUPS ships with quite different raster drivers processing CUPS raster. On my system I find in `/usr/lib/cups/filter/` these: `rastertoalps`, `rastertobj`, `rastertoepson`, `rastertoescp`, `rastertopcl`, `rastertoturboprint`, `rastertoapdk`, `rastertodymo`, `rastertoescp`, `rastertohp`, and `rastertoprinter`. Don't worry if you have less than this; some of these are installed by commercial add-ons to CUPS (like `rastertoturboprint`), others (like `rastertoprinter`) by third-party driver development projects (such as Gimp-Print) wanting to cooperate as closely as possible with CUPS.

18.5.9 CUPS Backends

The last part of any CUPS filtering chain is a backend. Backends are special programs that send the print-ready file to the final device. There is a separate backend program for any transfer protocol of sending printjobs over the network, or for every local interface. Every CUPS print queue needs to have a CUPS “*device-URI*” associated with it. The device URI is the way to encode the backend used to send the job to its destination. Network device-URIs are using two slashes in their syntax, local device URIs only one, as you can see from the following list. Keep in mind that local interface names may vary much from my examples, if your OS is not Linux:

usb — This backend sends printfiles to USB-connected printers. An example for the CUPS device-URI to use is: `usb:/dev/usb/lp0`.

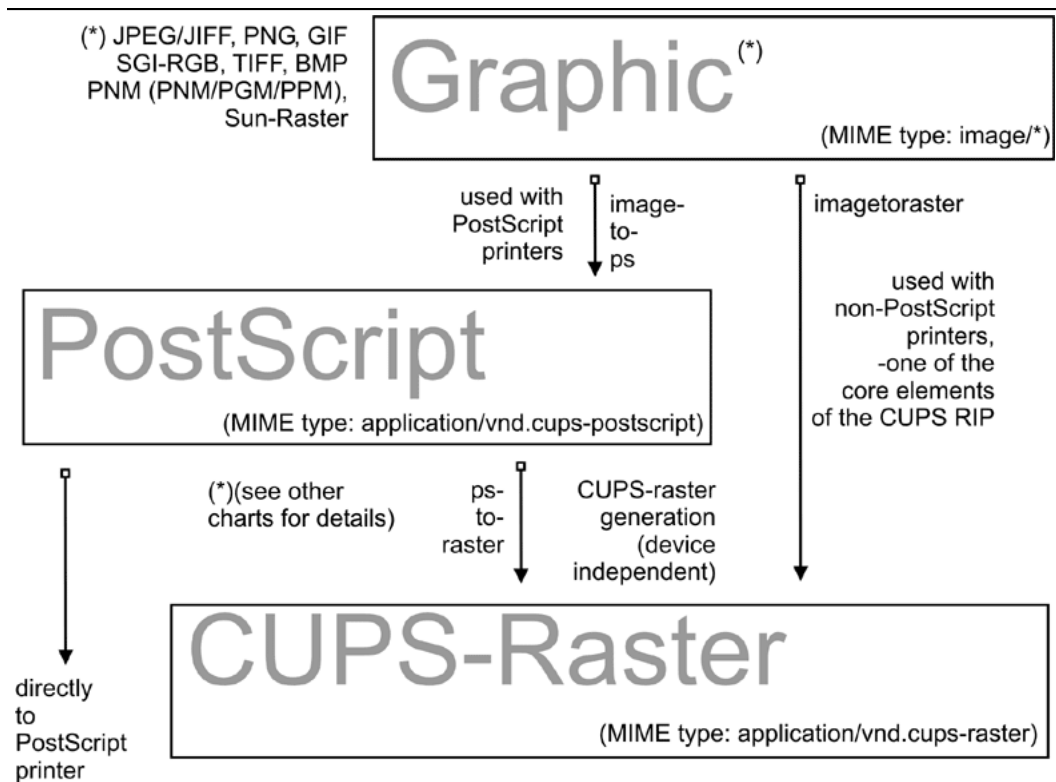


Figure 18.8. Image format to CUPS-raster format conversion.

serial — This backend sends printfiles to serially connected printers. An example for the CUPS device-URI to use is: `serial:/dev/ttyS0?baud=11500`.

parallel — This backend sends printfiles to printers connected to the parallel port. An example for the CUPS device-URI to use is: `parallel:/dev/lp0`.

scsi — This backend sends printfiles to printers attached to the SCSI interface. An example for the CUPS device-URI to use is: `scsi:/dev/sr1`.

lpd — This backend sends printfiles to LPR/LPD connected network printers. An example for the CUPS device-URI to use is: `lpd://remote_host_name/remote_queue_name`.

AppSocket/HP JetDirect — This backend sends printfiles to AppSocket (a.k.a. "HP JetDirect") connected network printers. An example for the CUPS device-URI to use is: `socket://10.11.12.13:9100`.

ipp — This backend sends printfiles to IPP connected network printers (or to other CUPS

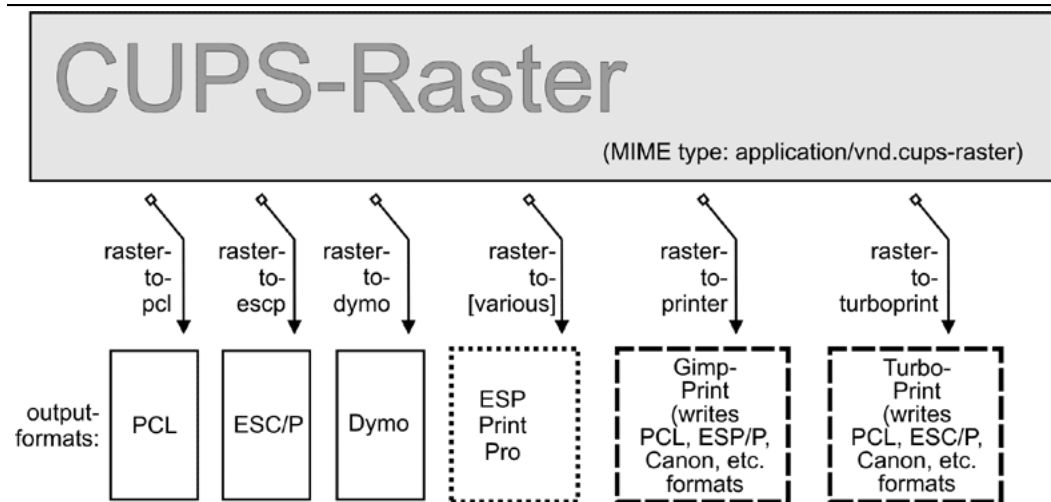


Figure 18.9. Raster to printer-specific formats.

servers). Examples for CUPS device-URIs to use are: `ipp://192.193.194.195/ipp` (for many HP printers) or `ipp://remote_cups_server/printers/remote_printer_name`.

http — This backend sends printfiles to HTTP connected printers. (The `http://` CUPS backend is only a symlink to the `ipp://` backend.) Examples for the CUPS device-URIs to use are: `http://192.193.194.195:631/ipp` (for many HP printers) or `http://remote_cups_server:631/printers/remote_printer_name`.

smb — This backend sends printfiles to printers shared by a Windows host. An example for CUPS device-URIs that may be used includes:

```
smb://workgroup/server/printerssharename
smb://server/printerssharename
smb://username:password@workgroup/server/printerssharename
smb://username:password@server/printerssharename
```

The `smb://` backend is a symlink to the Samba utility `smbspool` (does not ship with CUPS). If the symlink is not present in your CUPS backend directory, have your root user create it: `ln -s 'which smbspool' /usr/lib/cups/backend/smb`.

It is easy to write your own backends as shell or Perl scripts, if you need any modification or extension to the CUPS print system. One reason could be that you want to create “*special*” printers that send the printjobs as email (through a “*mailto:*” backend), convert them to PDF (through a “*pdftgen:*” backend) or dump them to “*/dev/null*”. (In fact I have the system-wide default printer set up to be connected to a `devnull:/` backend: there are just too many people sending jobs without specifying a printer, or scripts and programs which do not name a printer. The system-wide default deletes the job and sends a polite email back to the \$USER asking him to always specify the correct printer name.)

Not all of the mentioned backends may be present on your system or usable (depending on your hardware configuration). One test for all available CUPS backends is provided by the *lpinfo* utility. Used with the `-v` parameter, it lists all available backends:

```
$ lpinfo -v
```

18.5.10 The Role of *cupsonomatic*/*foomatic*

cupsonomatic filters may be the most widely used on CUPS installations. You must be clear about the fact that these were not developed by the CUPS people. They are a third party add-on to CUPS. They utilize the traditional Ghostscript devices to render jobs for CUPS. When troubleshooting, you should know about the difference. Here the whole rendering process is done in one stage, inside Ghostscript, using an appropriate device for the target printer. *cupsonomatic* uses PPDs that are generated from the Foomatic Printer & Driver Database at [Linuxprinting.org](http://linuxprinting.org).

You can recognize these PPDs from the line calling the *cupsonomatic* filter:

```
*cupsonomaticFilter: "application/vnd.cups-postscript 0 cupsonomatic"
```

You may find this line among the first 40 or so lines of the PPD file. If you have such a PPD installed, the printer shows up in the CUPS Web interface with a *foomatic* namepart for the driver description. *cupsonomatic* is a Perl script that runs Ghostscript with all the complicated command line options auto-constructed from the selected PPD and command line options give to the printjob.

However, *cupsonomatic* is now deprecated. Its PPDs (especially the first generation of them, still in heavy use out there) are not meeting the Adobe specifications. You might also suffer difficulties when you try to download them with “*Point’n’Print*” to Windows clients. A better and more powerful successor is now in a stable beta-version: it is called *foomatic-rip*. To use *foomatic-rip* as a filter with CUPS, you need the new-type PPDs. These have a similar but different line:

```
*cupsonomaticFilter: "application/vnd.cups-postscript 0 foomatic-rip"
```

The PPD generating engine at [Linuxprinting.org](http://linuxprinting.org) has been revamped. The new PPDs comply to the Adobe spec. On top, they also provide a new way to specify different quality levels (hi-res photo, normal color, grayscale, and draft) with a single click, whereas before you could have required five or more different selections (media type, resolution, inktype and dithering algorithm). There is support for custom-size media built in. There is support to switch print-options from page to page in the middle of a job. And the best thing is the new *foomatic-rip* now works seamlessly with all legacy spoolers too (like LPRng, BSD-LPD, PDQ, PPR and so on), providing for them access to use PPDs for their printing.

18.5.11 The Complete Picture

If you want to see an overview of all the filters and how they relate to each other, the complete picture of the puzzle is at the end of this document.

18.5.12 mime.convs

CUPS auto-constructs all possible filtering chain paths for any given MIME type, and every printer installed. But how does it decide in favor or against a specific alternative? (There may often be cases where there is a choice of two or more possible filtering chains for the same target printer.) Simple. You may have noticed the figures in the third column of the mime.convs file. They represent virtual costs assigned to this filter. Every possible filtering chain will sum up to a total “*filter cost*.” CUPS decides for the most “*inexpensive*” route.

TIP



The setting of *FilterLimit 1000* in *cupsd.conf* will not allow more filters to run concurrently than will consume a total of 1000 virtual filter cost. This is an efficient way to limit the load of any CUPS server by setting an appropriate “*FilterLimit*” value. A *FilterLimit* of 200 allows roughly one job at a time, while a *FilterLimit* of 1000 allows approximately five jobs maximum at a time.

18.5.13 “Raw” Printing

You can tell CUPS to print (nearly) any file “*raw*”. “*Raw*” means it will not be filtered. CUPS will send the file to the printer “*as is*” without bothering if the printer is able to digest it. Users need to take care themselves that they send sensible data formats only. Raw printing can happen on any queue if the “*-o raw*” option is specified on the command line. You can also set up raw-only queues by simply not associating any PPD with it. This command:

```
$ lpadmin -P rawprinter -v socket://11.12.13.14:9100 -E
```

sets up a queue named “*rawprinter*”, connected via the “*socket*” protocol (a.k.a. “*HP JetDirect*”) to the device at IP address 11.12.13.14, using port 9100. (If you had added a PPD with **-P /path/to/PPD** to this command line, you would have installed a “*normal*” print queue.

CUPS will automatically treat each job sent to a queue as a “*raw*” one, if it can’t find a PPD associated with the queue. However, CUPS will only send known MIME types (as defined in its own mime.types file) and refuse others.

18.5.14 application/octet-stream Printing

Any MIME type with no rule in the `/etc/cups/mime.types` file is regarded as unknown or *application/octet-stream* and will not be sent. Because CUPS refuses to print unknown MIME types per default, you will probably have experienced the fact that print jobs originating from Windows clients were not printed. You may have found an error message in your CUPS logs like:

```
Unable to convert file 0 to printable format for job
```

To enable the printing of *application/octet-stream* files, edit these two files:

- `/etc/cups/mime.convs`
- `/etc/cups/mime.types`

Both contain entries (at the end of the respective files) which must be uncommented to allow RAW mode operation for *application/octet-stream*. In `/etc/cups/mime.types` make sure this line is present:

```
application/octet-stream
```

This line (with no specific auto-typing rule set) makes all files not otherwise auto-typed a member of *application/octet-stream*. In `/etc/cups/mime.convs`, have this line:

```
application/octet-stream application/vnd.cups-raw 0 -
```

This line tells CUPS to use the *Null Filter* (denoted as “-”, doing nothing at all) on *application/octet-stream*, and tag the result as *application/vnd.cups-raw*. This last one is always a green light to the CUPS scheduler to now hand the file over to the backend connecting to the printer and sending it over.

NOTE



Editing the `mime.convs` and the `mime.types` file does not *enforce* “raw” printing, it only *allows* it.

CUPS being a more security-aware printing system than traditional ones does not by default allow one to send deliberate (possibly binary) data to printing devices. (This could be easily abused to launch a Denial of Service attack on your printer(s), causing at least the loss of a lot of paper and ink...) “Unknown” data are regarded by CUPS as *MIME type application/octet-stream*. While you *can* send data “raw”, the MIME type for these must be one that is known to CUPS and an allowed one. The file `/etc/cups/mime.types` defines the “rules” of how CUPS recognizes MIME types. The file `/etc/cups/mime.convs` decides which file conversion filter(s) may be applied to which MIME types.

18.5.15 PostScript Printer Descriptions (PPDs) for Non-PS Printers

Originally PPDs were meant to be used for PostScript printers only. Here, they help to send device-specific commands and settings to the RIP which processes the jobfile. CUPS has extended this scope for PPDs to cover non-PostScript printers too. This was not difficult, because it is a standardized file format. In a way it was logical too: CUPS handles PostScript and uses a PostScript RIP (Ghostscript) to process the jobfiles. The only difference is: a PostScript printer has the RIP built-in, for other types of printers the Ghostscript RIP runs on the host computer.

PPDs for a non-PS printer have a few lines that are unique to CUPS. The most important one looks similar to this:

```
*cupsFilter: application/vnd.cups-raster 66 rastertoprinter
```

It is the last piece in the CUPS filtering puzzle. This line tells the CUPS daemon to use as a last filter *rastertoprinter*. This filter should be served as input an *application/vnd.cups-raster* MIME type file. Therefore, CUPS should auto-construct a filtering chain, which delivers as its last output the specified MIME type. This is then taken as input to the specified *rastertoprinter* filter. After this the last filter has done its work (*rastertoprinter* is a Gimp-Print filter), the file should go to the backend, which sends it to the output device.

CUPS by default ships only a few generic PPDs, but they are good for several hundred printer models. You may not be able to control different paper trays, or you may get larger margins than your specific model supports. See Table 18.1 for summary information.

Table 18.1. PPDs shipped with CUPS

PPD file	Printer type
deskjet.ppd	older HP inkjet printers and compatible
deskjet2.ppd	newer HP inkjet printers and compatible
dymo.ppd	label printers
epson9.ppd	Epson 24pin impact printers and compatible
epson24.ppd	Epson 24pin impact printers and compatible
okidata9.ppd	Okidata 9pin impact printers and compatible
okidat24.ppd	Okidata 24pin impact printers and compatible
stcolor.ppd	older Epson Stylus Color printers
stcolor2.ppd	newer Epson Stylus Color printers
stphoto.ppd	older Epson Stylus Photo printers
stphoto2.ppd	newer Epson Stylus Photo printers
laserjet.ppd	all PCL printers. Further below is a discussion of several other driver/PPD-packages suitable for use with CUPS.

18.5.16 *cupsomatic/foomatic-rip* Versus native CUPS Printing

Native CUPS rasterization works in two steps:

- First is the *pstoraster* step. It uses the special CUPS device from ESP Ghostscript 7.05.x as its tool.
- Second comes the *rasterdriver* step. It uses various device-specific filters; there are several vendors who provide good quality filters for this step. Some are free software, some are shareware/non-free and some are proprietary.

Often this produces better quality (and has several more advantages) than other methods.

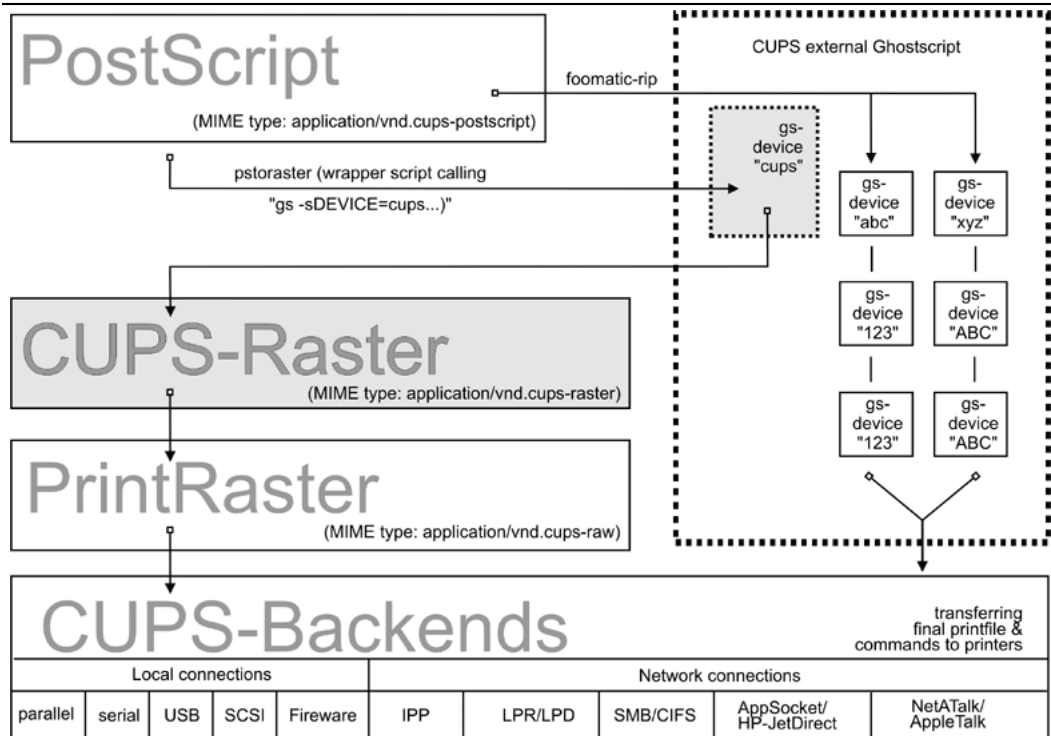


Figure 18.10. cupsomatic/foomatic Processing versus Native CUPS.

One other method is the *cupsomatic/foomatic-rip* way. Note that *cupsomatic* is *not* made by the CUPS developers. It is an independent contribution to printing development, made by people from Linuxprinting.org⁵. *cupsomatic* is no longer developed and maintained and is no longer supported. It has now been replaced by *foomatic-rip*. *foomatic-rip* is a complete re-write of the old *cupsomatic* idea, but very much improved and generalized to other (non-CUPS) spoolers. An upgrade to *foomatic-rip* is strongly advised, especially if you are upgrading to a recent version of CUPS, too.

Both the *cupsomatic* (old) and the *foomatic-rip* (new) methods from Linuxprinting.org use the traditional Ghostscript print file processing, doing everything in a single step. It therefore relies on all the other devices built into Ghostscript. The quality is as good (or bad) as Ghostscript rendering is in other spoolers. The advantage is that this method supports many printer models not supported (yet) by the more modern CUPS method.

⁵see also <http://www.cups.org/cups-help.html>

Of course, you can use both methods side by side on one system (and even for one printer, if you set up different queues) and find out which works best for you.

cupsomatic kidnaps the printfile after the *application/vnd.cups-postscript* stage and deviates it through the CUPS-external, system-wide Ghostscript installation. Therefore the printfile bypasses the *pstoraster* filter (and also bypasses the CUPS-raster-drivers *rastertosomething*). After Ghostscript finished its rasterization, *cupsomatic* hands the rendered file directly to the CUPS backend. The flowchart in Figure 18.10 illustrates the difference between native CUPS rendering and the *Foomatic/cupsomatic* method.

18.5.17 Examples for Filtering Chains

Here are a few examples of commonly occurring filtering chains to illustrate the workings of CUPS.

Assume you want to print a PDF file to an HP JetDirect-connected PostScript printer, but you want to print the pages 3-5, 7, 11-13 only, and you want to print them “two-up” and “duplex”:

- Your print options (page selection as required, two-up, duplex) are passed to CUPS on the command line.
- The (complete) PDF file is sent to CUPS and autotyped as *application/pdf*.
- The file therefore must first pass the *pdftops* pre-filter, which produces PostScript MIME type *application/postscript* (a preview here would still show all pages of the original PDF).
- The file then passes the *pstops* filter that applies the command line options: it selects the pages 2-5, 7 and 11-13, creates an imposed layout “2 pages on 1 sheet” and inserts the correct “duplex” command (as defined in the printer’s PPD) into the new PostScript file; the file is now of PostScript MIME type *application/vnd.cups-postscript*.
- The file goes to the *socket* backend, which transfers the job to the printers.

The resulting filter chain, therefore, is as drawn in Figure 18.11.

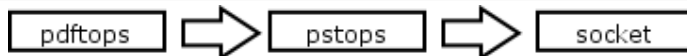


Figure 18.11. PDF to socket chain.

Assume your want to print the same filter to an USB-connected Epson Stylus Photo printer installed with the CUPS *stphoto2.ppd*. The first few filtering stages are nearly the same:

- Your print options (page selection as required, two-up, duplex) are passed to CUPS on the commandline.
- The (complete) PDF file is sent to CUPS and autotyped as *application/pdf*.

- The file must first pass the *pdftops* pre-filter, which produces PostScript MIME type *application/postscript* (a preview here would still show all pages of the original PDF).
- The file then passes the “*pstops*” filter that applies the commandline options: it selects the pages 2-5, 7 and 11-13, creates an imposed layout “*two pages on one sheet*” and inserts the correct “*duplex*” command... (Oops — this printer and PPD do not support duplex printing at all — so this option will be ignored) into the new PostScript file; the file is now of PostScript MIME type *application/vnd.cups-postscript*.
- The file then passes the *pstoraster* stage and becomes MIME type *application/cups-raster*.
- Finally, the *rastertoepson* filter does its work (as indicated in the printer’s PPD), creating the rinter-specific raster data and embedding any user-selected print-options into the print data stream.
- The file goes to the *usb* backend, which transfers the job to the printers.

The resulting filter chain therefore is as drawn in Figure 18.12.

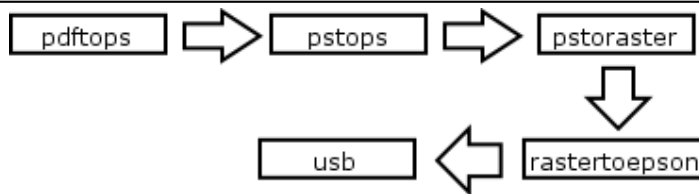


Figure 18.12. PDF to USB chain.

18.5.18 Sources of CUPS Drivers/PPDs

On the Internet you can now find many thousands of CUPS-PPD files (with their companion filters), in many national languages supporting more than thousand non-PostScript models.

- ESP PrintPro⁶ (commercial, non-free) is packaged with more than three thousand PPDs, ready for successful use “*out of the box*” on Linux, Mac OS X, IBM-AIX, HP-UX, Sun-Solaris, SGI-IRIX, Compaq Tru64, Digital UNIX, and some more commercial Unices (it is written by the CUPS developers themselves and its sales help finance the further development of CUPS, as they feed their creators).
- The Gimp-Print-Project⁷ (GPL, free software) provides around 140 PPDs (supporting nearly 400 printers, many driven to photo quality output), to be used alongside the Gimp-Print CUPS filters.
- TurboPrint⁸ (shareware, non-free) supports roughly the same amount of printers in excellent quality.

⁶<http://www1.easysw.com/printpro/>

⁷<http://gimp-print.sourceforge.net/>

⁸<http://www.turboprint.com/>

- OMNI⁹ (LPGL, free) is a package made by IBM, now containing support for more than 400 printers, stemming from the inheritance of IBM OS/2 Know-How ported over to Linux (CUPS support is in a beta-stage at present).
- HPIJS¹⁰ (BSD-style licenses, free) supports around 150 of HP's own printers and is also providing excellent print quality now (currently available only via the Foomatic path).
- Foomatic/cupsomatic¹¹ (LPGL, free) from [Linuxprinting.org](http://linuxprinting.org) are providing PPDs for practically every Ghostscript filter known to the world (including Omni, Gimp-Print and HPIJS).

18.5.19 Printing with Interface Scripts

CUPS also supports the usage of “*interface scripts*” as known from System V AT&T printing systems. These are often used for PCL printers, from applications that generate PCL print jobs. Interface scripts are specific to printer models. They have a similar role as PPDs for PostScript printers. Interface scripts may inject the Escape sequences as required into the print data stream, if the user has chosen to select a certain paper tray, or print landscape, or use A3 paper, etc. Interfaces scripts are practically unknown in the Linux realm. On HP-UX platforms they are more often used. You can use any working interface script on CUPS too. Just install the printer with the `-i` option:

```
root# lpadmin -p pclprinter -v socket://11.12.13.14:9100 \  
-i /path/to/interface-script
```

Interface scripts might be the “*unknown animal*” to many. However, with CUPS they provide the easiest way to plug in your own custom-written filtering script or program into one specific print queue (some information about the traditional usage of interface scripts is to be found at <http://playground.sun.com/printing/documentation/interface.html>).

18.6 Network Printing (Purely Windows)

Network printing covers a lot of ground. To understand what exactly goes on with Samba when it is printing on behalf of its Windows clients, let's first look at a “*purely Windows*” setup: Windows clients with a Windows NT print server.

18.6.1 From Windows Clients to an NT Print Server

Windows clients printing to an NT-based print server have two options. They may:

- Execute the driver locally and render the GDI output (EMF) into the printer-specific format on their own.

⁹<http://www-124.ibm.com/developerworks/oss/linux/projects/omni/>

¹⁰<http://hpinkjet.sourceforge.net/>

¹¹<http://www.linuxprinting.org/>

- Send the GDI output (EMF) to the server, where the driver is executed to render the printer specific output.

Both print paths are shown in the flowcharts in Figure 18.13 and Figure 18.14.

18.6.2 Driver Execution on the Client

In the first case the print server must spool the file as raw, meaning it shouldn't touch the jobfile and try to convert it in any way. This is what a traditional UNIX-based print server can do too, and at a better performance and more reliably than an NT print server. This is what most Samba administrators probably are familiar with. One advantage of this setup is that this “*spooling-only*” print server may be used even if no driver(s) for UNIX are available if it is sufficient to have the Windows client drivers available; and installed on the clients.

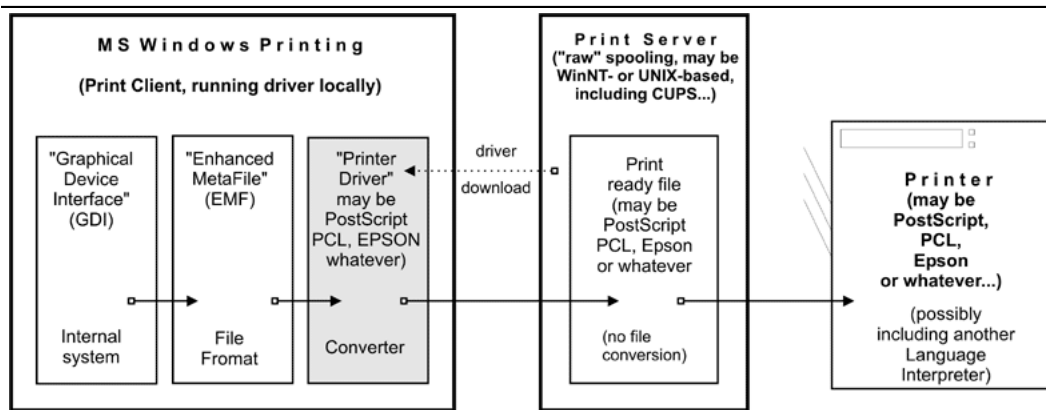


Figure 18.13. Print driver execution on the client.

18.6.3 Driver Execution on the Server

The other path executes the printer driver on the server. The client transfers print files in EMF format to the server. The server uses the PostScript, PCL, ESC/P or other driver to convert the EMF file into the printer-specific language. It is not possible for UNIX to do the same. Currently, there is no program or method to convert a Windows client's GDI output on a UNIX server into something a printer could understand.

However, there is something similar possible with CUPS. Read on.

18.7 Network Printing (Windows Clients — UNIX/Samba Print Servers)

Since UNIX print servers *cannot* execute the Win32 program code on their platform, the picture is somewhat different. However, this does not limit your options all that much. On the contrary, you may have a way here to implement printing features that are not possible otherwise.

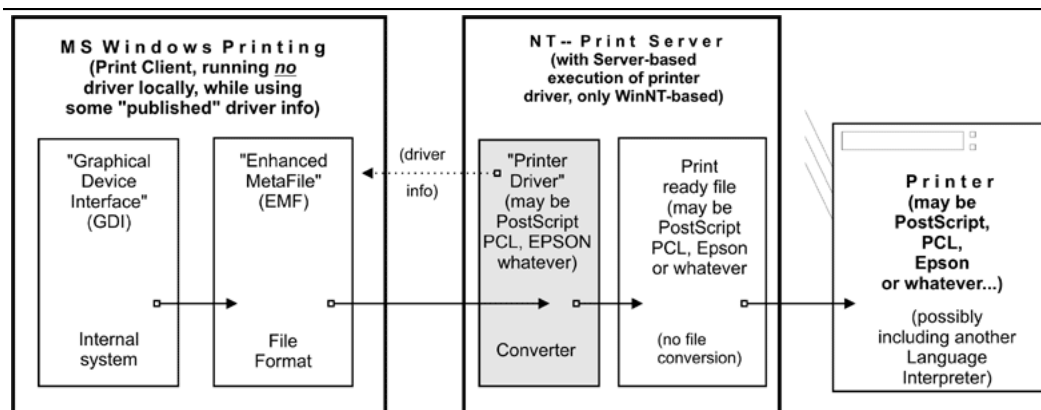


Figure 18.14. Print driver execution on the server.

18.7.1 From Windows Clients to a CUPS/Samba Print Server

Here is a simple recipe showing how you can take advantage of CUPS' powerful features for the benefit of your Windows network printing clients:

- Let the Windows clients send PostScript to the CUPS server.
- Let the CUPS server render the PostScript into device-specific raster format.

This requires the clients to use a PostScript driver (even if the printer is a non-PostScript model. It also requires that you have a driver on the CUPS server.

First, to enable CUPS-based printing through Samba the following options should be set in your `smb.conf` file [global] section:

```
printing = cups
printcap = cups
```

When these parameters are specified, all manually set print directives (like `print command`, or `lppause command`) in `smb.conf` (as well as in Samba itself) will be ignored. Instead, Samba will directly interface with CUPS through its application program interface (API), as long as Samba has been compiled with CUPS library (`libcups`) support. If Samba has not been compiled with CUPS support, and if no other print commands are set up, then printing will use the `System V AT&T` command set, with the `-oraw` option automatically passing through (if you want your own defined print commands to work with a Samba that has CUPS support compiled in, simply use `printing = sysv`).

18.7.2 Samba Receiving Jobfiles and Passing Them to CUPS

Samba *must* use its own spool directory (it is set by a line similar to `path = /var/spool/samba`, in the `[printers]` or `[printername]` section of `smb.conf`). Samba receives the job in its own spool space and passes it into the spool directory of CUPS (the CUPS spooling directory is set by the `RequestRoot` directive, in a line that defaults to `RequestRoot /var/spool/cups`). CUPS checks the access rights of its spool dir and resets it to healthy

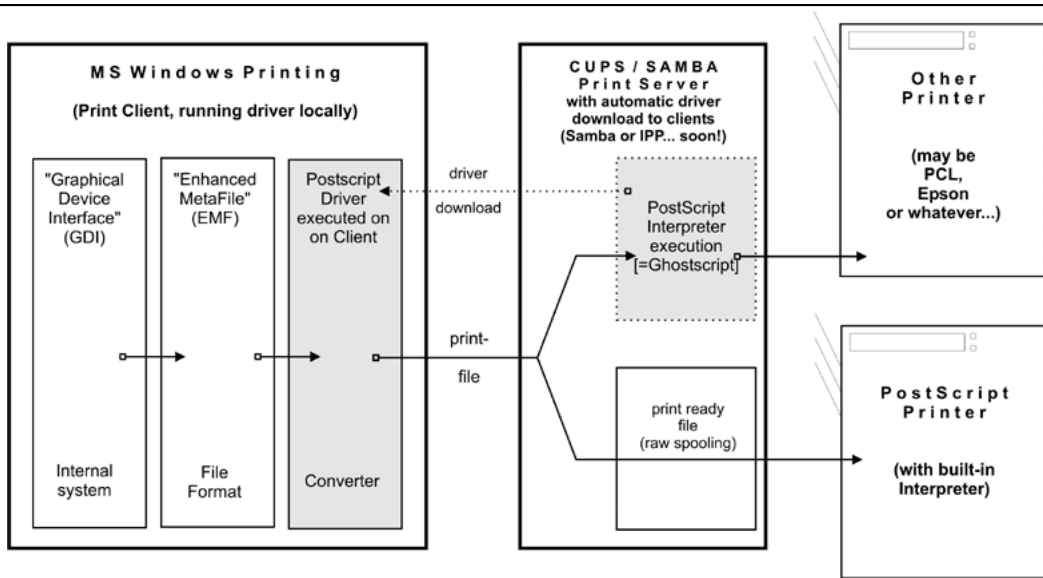


Figure 18.15. Printing via CUPS/Samba server.

values with every restart. We have seen quite a few people who had used a common spooling space for Samba and CUPS, and were struggling for weeks with this “*problem.*”

A Windows user authenticates only to Samba (by whatever means is configured). If Samba runs on the same host as CUPS, you only need to allow “*localhost*” to print. If they run on different machines, you need to make sure the Samba host gets access to printing on CUPS.

18.8 Network PostScript RIP

This section discusses the use of CUPS filters on the server — configuration where clients make use of a PostScript driver with CUPS-PPDs.

PPDs can control all print device options. They are usually provided by the manufacturer, if you own a PostScript printer, that is. PPD files (PostScript Printer Descriptions) are always a component of PostScript printer drivers on MS Windows or Apple Mac OS systems. They are ASCII files containing user-selectable print options, mapped to appropriate PostScript, PCL or PJI commands for the target printer. Printer driver GUI dialogs translate these options “*on-the-fly*” into buttons and drop-down lists for the user to select.

CUPS can load, without any conversions, the PPD file from any Windows (NT is recommended) PostScript driver and handle the options. There is a Web browser interface to the print options (select <http://localhost:631/printers/> and click on one **Configure Printer** button to see it), or a command line interface (see **man lpoptions** or see if you have **lphelp** on your system). There are also some different GUI frontends on Linux/UNIX, which can present PPD options to users. PPD options are normally meant to be evaluated by the PostScript RIP on the real PostScript printer.

18.8.1 PPDs for Non-PS Printers on UNIX

CUPS does not limit itself to “*real*” PostScript printers in its usage of PPDs. The CUPS developers have extended the scope of the PPD concept to also describe available device and driver options for non-PostScript printers through CUPS-PPDs.

This is logical, as CUPS includes a fully featured PostScript interpreter (RIP). This RIP is based on Ghostscript. It can process all received PostScript (and additionally many other file formats) from clients. All CUPS-PPDs geared to non-PostScript printers contain an additional line, starting with the keyword **cupsFilter*. This line tells the CUPS print system which printer-specific filter to use for the interpretation of the supplied PostScript. Thus CUPS lets all its printers appear as PostScript devices to its clients, because it can act as a PostScript RIP for those printers, processing the received PostScript code into a proper raster print format.

18.8.2 PPDs for Non-PS Printers on Windows

CUPS-PPDs can also be used on Windows-Clients, on top of a “*core*” PostScript driver (now recommended is the “CUPS PostScript Driver for WindowsNT/200x/XP”; you can also use the Adobe one, with limitations). This feature enables CUPS to do a few tricks no other spooler can do:

- Act as a networked PostScript RIP (Raster Image Processor), handling printfiles from all client platforms in a uniform way.
- Act as a central accounting and billing server, since all files are passed through the `pstops` filter and are, therefore, logged in the CUPS `page_log` file. *Note:* this cannot happen with “*raw*” print jobs, which always remain unfiltered per definition.
- Enable clients to consolidate on a single PostScript driver, even for many different target printers.

Using CUPS PPDs on Windows clients enables these to control all print job settings just as a UNIX client can do.

18.9 Windows Terminal Servers (WTS) as CUPS Clients

This setup may be of special interest to people experiencing major problems in WTS environments. WTS often need a multitude of non-PostScript drivers installed to run their clients’ variety of different printer models. This often imposes the price of much increased instability.

18.9.1 Printer Drivers Running in “*Kernel Mode*” Cause Many Problems

In Windows NT printer drivers which run in “*Kernel Mode*”, introduces a high risk for the stability of the system if the driver is not really stable and well-tested. And there are a lot of bad drivers out there! Especially notorious is the example of the PCL printer driver that had an additional sound module running, to notify users via soundcard of their finished

jobs. Do I need to say that this one was also reliably causing “*blue screens of death*” on a regular basis?

PostScript drivers are generally well tested. They are not known to cause any problems, even though they also run in kernel mode. This might be because there have been so far only two different PostScript drivers: the ones from Adobe and the one from Microsoft. Both are well tested and are as stable as you can imagine on Windows. The CUPS driver is derived from the Microsoft one.

18.9.2 Workarounds Impose Heavy Limitations

In many cases, in an attempt to work around this problem, site administrators have resorted to restricting the allowed drivers installed on their WTS to one generic PCL and one PostScript driver. This, however, restricts the clients in the number of printer options available for them. Often they can't get out more than simplex prints from one standard paper tray, while their devices could do much better, if driven by a different driver!

18.9.3 CUPS: A “*Magical Stone*”?

Using a PostScript driver, enabled with a CUPS-PPD, seems to be a very elegant way to overcome all these shortcomings. There are, depending on the version of Windows OS you use, up to three different PostScript drivers available: Adobe, Microsoft and CUPS PostScript drivers. None of them is known to cause major stability problems on WTS (even if used with many different PPDs). The clients will be able to (again) chose paper trays, duplex printing and other settings. However, there is a certain price for this too: a CUPS server acting as a PostScript RIP for its clients requires more CPU and RAM than when just acting as a “*raw spooling*” device. Plus, this setup is not yet widely tested, although the first feedbacks look very promising.

18.9.4 PostScript Drivers with No Major Problems — Even in Kernel Mode

More recent printer drivers on W200x and XP no longer run in kernel mode (unlike Windows NT). However, both operating systems can still use the NT drivers, running in kernel mode (you can roughly tell which is which as the drivers in subdirectory “*2*” of “*W32X86*” are “*old*” ones). As was said before, the Adobe as well as the Microsoft PostScript drivers are not known to cause any stability problems. The CUPS driver is derived from the Microsoft one. There is a simple reason for this: The MS DDK (Device Development Kit) for Windows NT (which used to be available at no cost to licensees of Visual Studio) includes the source code of the Microsoft driver, and licensees of Visual Studio are allowed to use and modify it for their own driver development efforts. This is what the CUPS people have done. The license does not allow them to publish the whole of the source code. However, they have released the “*diff*” under the GPL, and if you are the owner of an “*MS DDK for Windows NT*,” you can check the driver yourself.

18.10 Configuring CUPS for Driver Download

As we have said before, all previously known methods to prepare client printer drivers on the Samba server for download and Point'n'Print convenience of Windows workstations are working with CUPS, too. These methods were described in the previous chapter. In reality, this is a pure Samba business and only relates to the Samba/Windows client relationship.

18.10.1 *cupsaddsmb*: The Unknown Utility

The **cupsaddsmb** utility (shipped with all current CUPS versions) is an alternate method to transfer printer drivers into the Samba *[print\$]* share. Remember, this share is where clients expect drivers deposited and setup for download and installation. It makes the sharing of any (or all) installed CUPS printers quite easy. **cupsaddsmb** can use the Adobe PostScript driver as well as the newly developed CUPS PostScript Driver for Windows NT/200x/XP. *cupsaddsmb* does *not* work with arbitrary vendor printer drivers, but only with the *exact* driver files that are named in its man page.

The CUPS printer driver is available from the CUPS download site. Its package name is `cups-samba-[version].tar.gz`. It is preferred over the Adobe drivers since it has a number of advantages:

- It supports a much more accurate page accounting.
- It supports banner pages, and page labels on all printers.
- It supports the setting of a number of job IPP attributes (such as job-priority, page-label and job-billing).

However, currently only Windows NT, 2000 and XP are supported by the CUPS drivers. You will also need to get the respective part of Adobe driver if you need to support Windows 95, 98 and ME clients.

18.10.2 Prepare Your `smb.conf` for *cupsaddsmb*

Prior to running **cupsaddsmb**, you need the settings in `smb.conf` as shown in Example 18.3:

18.10.3 CUPS “*PostScript Driver for Windows NT/200x/XP*”

CUPS users may get the exact same packages from <http://www.cups.org/software.html>. It is a separate package from the CUPS base software files, tagged as CUPS 1.1.x Windows NT/200x/XP Printer Driver for Samba (tar.gz, 192k). The filename to download is `cups-samba-1.1.x.tar.gz`. Upon untar and unzipping, it will reveal these files:

```
root# tar xvzf cups-samba-1.1.19.tar.gz
cups-samba.install
cups-samba.license
cups-samba.readme
cups-samba.remove
```

Example 18.3. smb.conf for cupsaddsmb usage

```
[global]
load printers = yes
printing = cups
printcap name = cups

[printers]
comment = All Printers
path = /var/spool/samba
browseable = no
public = yes
# setting depends on your requirements
guest ok = yes
writable = no
printable = yes
printer admin = root

[print$]
comment = Printer Drivers
path = /etc/samba/drivers
browseable = yes
guest ok = no
read only = yes
write list = root
```

cups-samba.ss

These have been packaged with the ESP meta packager software EPM. The `*.install` and `*.remove` files are simple shell scripts, which untars the `*.ss` (the `*.ss` is nothing else but a tar-archive, which can be untarred by “`tar`” too). Then it puts the content into `/usr/share/cups/drivers/`. This content includes three files:

```
root# tar tv cups-samba.ss
cupsdrv.dll
cupsui.dll
cups.hlp
```

The `cups-samba.install` shell scripts are easy to handle:

```
root# ./cups-samba.install
[...]
Installing software...
Updating file permissions...
```

```
Running post-install commands...
Installation is complete.
```

The script should automatically put the driver files into the `/usr/share/cups/drivers/` directory.

WARNING



Due to a bug, one recent CUPS release puts the `cups.hlp` driver file into `/usr/share/drivers/` instead of `/usr/share/cups/drivers/`. To work around this, copy/move the file (after running the `./cups-samba.install` script) manually to the correct place.

```
root# cp /usr/share/drivers/cups.hlp /usr/share/cups/drivers/
```

This new CUPS PostScript driver is currently binary-only, but free of charge. No complete source code is provided (yet). The reason is that it has been developed with the help of the Microsoft Driver Developer Kit (DDK) and compiled with Microsoft Visual Studio 6. Driver developers are not allowed to distribute the whole of the source code as free software. However, CUPS developers released the “*diff*” in source code under the GPL, so anybody with a license of Visual Studio and a DDK will be able to compile for him/herself.

18.10.4 Recognizing Different Driver Files

The CUPS drivers do not support the older Windows 95/98/Me, but only the Windows NT/2000/XP client.

Windows NT, 2000 and XP are supported by:

- `cups.hlp`
- `cupsdrv.dll`
- `cupsui.dll`

Adobe drivers are available for the older Windows 95/98/Me as well as the Windows NT/2000/XP clients. The set of files is different from the different platforms.

Windows 95, 98 and ME are supported by:

- `ADFONT.SMF`
- `ADOBEP4.DRV`
- `ADOBEP4.HLP`
- `DEFPRTR2.PPD`
- `ICONLIB.DLL`

- PSMON.DLL

Windows NT, 2000 and XP are supported by:

- ADOBEPS5.DLL
- ADOBEPSU.DLL
- ADOBEPSU.HLP

NOTE



If both the Adobe driver files and the CUPS driver files for the support of Windows NT/200x/XP are present in `FIXME`, the Adobe ones will be ignored and the CUPS ones will be used. If you prefer — for whatever reason — to use Adobe-only drivers, move away the three CUPS driver files. The Windows 9x/Me clients use the Adobe drivers in any case.

18.10.5 Acquiring the Adobe Driver Files

Acquiring the Adobe driver files seems to be unexpectedly difficult for many users. They are not available on the Adobe Web site as single files and the self-extracting and/or self-installing `Windows-.exe` is not easy to locate either. Probably you need to use the included native installer and run the installation process on one client once. This will install the drivers (and one Generic PostScript printer) locally on the client. When they are installed, share the Generic PostScript printer. After this, the client's `[print$]` share holds the Adobe files, from where you can get them with `smbclient` from the CUPS host.

18.10.6 ESP Print Pro PostScript Driver for Windows NT/200x/XP

Users of the ESP Print Pro software are able to install their Samba drivers package for this purpose with no problem. Retrieve the driver files from the normal download area of the ESP Print Pro software at <http://www.easysw.com/software.html>. You need to locate the link labelled “*SAMBA*” among the **Download Printer Drivers for ESP Print Pro 4.x** area and download the package. Once installed, you can prepare any driver by simply highlighting the printer in the Printer Manager GUI and select **Export Driver...** from the menu. Of course you need to have prepared Samba beforehand to handle the driver files; i.e., setup the `[print$]` share, and so on. The ESP Print Pro package includes the CUPS driver files as well as a (licensed) set of Adobe drivers for the Windows 95/98/Me client family.

18.10.7 Caveats to be Considered

Once you have run the install script (and possibly manually moved the `cups.hlp` file to `/usr/share/cups/drivers/`), the driver is ready to be put into Samba's `[print$]` share (which often maps to `/etc/samba/drivers/` and contains a subdirectory tree with `WIN40`

and *W32X86* branches). You do this by running **cupsaddsmb** (see also **man cupsaddsmb** for CUPS since release 1.1.16).

TIP

You may need to put `root` into the `smbpasswd` file by running **smbpasswd**; this is especially important if you should run this whole procedure for the first time, and are not working in an environment where everything is configured for *single sign on* to a Windows Domain Controller.

Once the driver files are in the `[print$]` share and are initialized, they are ready to be downloaded and installed by the Windows NT/200x/XP clients.

NOTE

Win 9x/Me clients will not work with the CUPS PostScript driver. For these you still need to use the `ADOBE*.*` drivers as previously stated.

NOTE

It is not harmful if you still have the `ADOBE*.*` driver files from previous installations in the `/usr/share/cups/drivers/` directory. The new **cupsaddsmb** (from 1.1.16) will automatically prefer its own drivers if it finds both.

NOTE



Should your Windows clients have had the old `ADOBE*.*` files for the Adobe PostScript driver installed, the download and installation of the new CUPS PostScript driver for Windows NT/200x/XP will fail at first. You need to wipe the old driver from the clients first. It is not enough to “*delete*” the printer, as the driver files will still be kept by the clients and re-used if you try to re-install the printer. To really get rid of the Adobe driver files on the clients, open the **Printers** folder (possibly via **Start > Settings > Control Panel > Printers**), right-click on the folder background and select **Server Properties**. When the new dialog opens, select the **Drivers** tab. On the list select the driver you want to delete and click the **Delete** button. This will only work if there is not one single printer left that uses that particular driver. You need to “*delete*” all printers using this driver in the **Printers** folder first. You will need Administrator privileges to do this.

NOTE



Once you have successfully downloaded the CUPS PostScript driver to a client, you can easily switch all printers to this one by proceeding as described in Chapter 17, *Classical Printing Support*. Either change a driver for an existing printer by running the **Printer Properties** dialog, or use **rpcclient** with the **setdriver** subcommand.

18.10.8 Windows CUPS PostScript Driver Versus Adobe Driver

Are you interested in a comparison between the CUPS and the Adobe PostScript drivers? For our purposes these are the most important items that weigh in favor of the CUPS ones:

- No hassle with the Adobe EULA.
- No hassle with the question “*Where do I get the ADOBE*. * driver files from?*”
- The Adobe drivers (on request of the printer PPD associated with them) often put a PJI header in front of the main PostScript part of the print file. Thus, the printfile starts with `<1B >%-12345X` or `<escape>%-12345X` instead of `!PS`). This leads to the CUPS daemon auto-typing the incoming file as a print-ready file, not initiating a pass through the `pstops` filter (to speak more technically, it is not regarded as the generic MIME-type `application/postscript`, but as the more special MIME type `application/cups.vnd-postscript`), which therefore also leads to the page accounting in `/var/log/cups/page.log` not receiving the exact number of pages; instead the dummy page number of “1” is logged in a standard setup).

- The Adobe driver has more options to misconfigure the PostScript generated by it (like setting it inadvertently to **Optimize for Speed**, instead of **Optimize for Portability**, which could lead to CUPS being unable to process it).
- The CUPS PostScript driver output sent by Windows clients to the CUPS server is guaranteed to auto-type as the generic MIME type *application/postscript*, thus passing through the CUPS *pstops* filter and logging the correct number of pages in the *page_log* for accounting and quota purposes.
- The CUPS PostScript driver supports the sending of additional standard (IPP) print options by Windows NT/200x/XP clients. Such additional print options are: naming the CUPS standard *banner pages* (or the custom ones, should they be installed at the time of driver download), using the CUPS page-label option, setting a job-priority, and setting the scheduled time of printing (with the option to support additional useful IPP job attributes in the future).
- The CUPS PostScript driver supports the inclusion of the new **cupsJobTicket* comments at the beginning of the PostScript file (which could be used in the future for all sort of beneficial extensions on the CUPS side, but which will not disturb any other applications as they will regard it as a comment and simply ignore it).
- The CUPS PostScript driver will be the heart of the fully fledged CUPS IPP client for Windows NT/200x/XP to be released soon (probably alongside the first beta release for CUPS 1.2).

18.10.9 Run cupsaddsmb (Quiet Mode)

The **cupsaddsmb** command copies the needed files into your *[print\$]* share. Additionally, the PPD associated with this printer is copied from */etc/cups/ppd/* to *[print\$]*. There the files wait for convenient Windows client installations via Point'n'Print. Before we can run the command successfully, we need to be sure that we can authenticate toward Samba. If you have a small network, you are probably using user-level security (*security = user*).

Here is an example of a successfully run **cupsaddsmb** command:

```
root# cupsaddsmb -U root infotec_IS2027
Password for root required to access localhost via Samba: ['secret']
```

To share *all* printers and drivers, use the **-a** parameter instead of a printer name. Since **cupsaddsmb** “*exports*” the printer drivers to Samba, it should be obvious that it only works for queues with a CUPS driver associated.

18.10.10 Run cupsaddsmb with Verbose Output

Probably you want to see what’s going on. Use the **-v** parameter to get a more verbose output. The output below was edited for better readability: all “\” at the end of a line indicate that I inserted an artificial line break plus some indentation here:

WARNING



You will see the root password for the Samba account printed on screen.

```

root# cupsaddsmb -U root -v infotec_2105
Password for root required to access localhost via GANDALF:
Running command: smbclient //localhost/print\$ -N -U'root%secret' \
  -c 'mkdir W32X86; \
    put /var/spool/cups/tmp/3e98bf2d333b5 W32X86/infotec_2105.ppd; \
    put /usr/share/cups/drivers/cupsdrv.dll W32X86/cupsdrv.dll; \
    put /usr/share/cups/drivers/cupsui.dll W32X86/cupsui.dll; \
    put /usr/share/cups/drivers/cups.hlp W32X86/cups.hlp'
added interface ip=10.160.51.60 bcast=10.160.51.255 nmask=255.255.252.0
Domain=[CUPS-PRINT] OS=[UNIX] Server=[Samba 2.2.7a]
NT_STATUS_OBJECT_NAME_COLLISION making remote directory \W32X86
putting file /var/spool/cups/tmp/3e98bf2d333b5 as \W32X86/infotec_2105.ppd
putting file /usr/share/cups/drivers/cupsdrv.dll as \W32X86/cupsdrv.dll
putting file /usr/share/cups/drivers/cupsui.dll as \W32X86/cupsui.dll
putting file /usr/share/cups/drivers/cups.hlp as \W32X86/cups.hlp

Running command: rpcclient localhost -N -U'root%secret'
  -c 'adddriver "Windows NT x86" \
    "infotec_2105:cupsdrv.dll:infotec_2105.ppd:cupsui.dll:cups.hlp:NULL: \
    RAW:NULL"'
cmd = adddriver "Windows NT x86" \
  "infotec_2105:cupsdrv.dll:infotec_2105.ppd:cupsui.dll:cups.hlp:NULL: \
  RAW:NULL"
Printer Driver infotec_2105 successfully installed.

Running command: smbclient //localhost/print\$ -N -U'root%secret' \
  -c 'mkdir WIN40; \
    put /var/spool/cups/tmp/3e98bf2d333b5 WIN40/infotec_2105.PPD; \
    put /usr/share/cups/drivers/ADFFONTS.MFM WIN40/ADFFONTS.MFM; \
    put /usr/share/cups/drivers/ADOBEPS4.DRV WIN40/ADOBEPS4.DRV; \
    put /usr/share/cups/drivers/ADOBEPS4.HLP WIN40/ADOBEPS4.HLP; \
    put /usr/share/cups/drivers/DEFPRTR2.PPD WIN40/DEFPRTR2.PPD; \
    put /usr/share/cups/drivers/ICONLIB.DLL WIN40/ICONLIB.DLL; \
    put /usr/share/cups/drivers/PSMON.DLL WIN40/PSMON.DLL;'
added interface ip=10.160.51.60 bcast=10.160.51.255 nmask=255.255.252.0
Domain=[CUPS-PRINT] OS=[UNIX] Server=[Samba 2.2.7a]
NT_STATUS_OBJECT_NAME_COLLISION making remote directory \WIN40
putting file /var/spool/cups/tmp/3e98bf2d333b5 as \WIN40/infotec_2105.PPD
putting file /usr/share/cups/drivers/ADFFONTS.MFM as \WIN40/ADFFONTS.MFM

```

```

putting file /usr/share/cups/drivers/ADOBEPS4.DRV as \WIN40/ADOBEPS4.DRV
putting file /usr/share/cups/drivers/ADOBEPS4.HLP as \WIN40/ADOBEPS4.HLP
putting file /usr/share/cups/drivers/DEFPRTR2.PPD as \WIN40/DEFPRTR2.PPD
putting file /usr/share/cups/drivers/ICONLIB.DLL as \WIN40/ICONLIB.DLL
putting file /usr/share/cups/drivers/PSMON.DLL as \WIN40/PSMON.DLL

```

```

Running command: rpcclient localhost -N -U'root%secret' \
-c 'adddriver "Windows 4.0" \
"infotec_2105:ADOBEPS4.DRV:infotec_2105.PPD:NULL:ADOBEPS4.HLP: \
PSMON.DLL:RAW:ADOBEPS4.DRV,infotec_2105.PPD,ADOBEPS4.HLP,PSMON.DLL, \
ADFONTS.MFM,DEFPRTR2.PPD,ICONLIB.DLL"'
cmd = adddriver "Windows 4.0" "infotec_2105:ADOBEPS4.DRV:\
infotec_2105.PPD:NULL:ADOBEPS4.HLP:PSMON.DLL:RAW:ADOBEPS4.DRV,\
infotec_2105.PPD,ADOBEPS4.HLP,PSMON.DLL,ADFONTS.MFM,DEFPRTR2.PPD,\
ICONLIB.DLL"
Printer Driver infotec_2105 successfully installed.

```

```

Running command: rpcclient localhost -N -U'root%secret' \
-c 'setdriver infotec_2105 infotec_2105'
cmd = setdriver infotec_2105 infotec_2105
Successfully set infotec_2105 to driver infotec_2105.

```

If you look closely, you'll discover your root password was transferred unencrypted over the wire, so beware! Also, if you look further, you'll discover error messages like `NT_STATUS_OBJECT_NAME_COLLISION` in between. They occur, because the directories `WIN40` and `W32X86` already existed in the `[print$]` driver download share (from a previous driver installation). They are harmless here.

18.10.11 Understanding cupsaddsmb

What has happened? What did `cupsaddsmb` do? There are five stages of the procedure:

1. Call the CUPS server via IPP and request the driver files and the PPD file for the named printer.
2. Store the files temporarily in the local `TEMPDIR` (as defined in `cupsd.conf`).
3. Connect via `smbclient` to the Samba server's `[print$]` share and put the files into the share's `WIN40` (for Windows 9x/Me) and `W32X86/` (for Windows NT/200x/XP) subdirectories.
4. Connect via `rpcclient` to the Samba server and execute the `adddriver` command with the correct parameters.
5. Connect via `rpcclient` to the Samba server a second time and execute the `setdriver` command.

NOTE



You can run the **cupsaddsmb** utility with parameters to specify one remote host as Samba host and a second remote host as CUPS host. Especially if you want to get a deeper understanding, it is a good idea to try it and see more clearly what is going on (though in real life most people will have their CUPS and Samba servers run on the same host):

```
root# cupsaddsmb -H sambaserver -h cupserver -v printer
```

18.10.12 How to Recognize If cupsaddsmb Completed Successfully

You *must* always check if the utility completed successfully in all fields. You need as a minimum these three messages among the output:

1. *Printer Driver infotec_2105 successfully installed. #* (for the W32X86 == Windows NT/200x/XP architecture).
2. *Printer Driver infotec_2105 successfully installed. #* (for the WIN40 == Windows 9x/Me architecture).
3. *Successfully set [printerXPZ] to driver [printerXYZ].*

These messages are probably not easily recognized in the general output. If you run **cupsaddsmb** with the **-a** parameter (which tries to prepare *all* active CUPS printer drivers for download), you might miss if individual printers drivers had problems installing properly. Here a redirection of the output will help you analyze the results in retrospective.

NOTE



It is impossible to see any diagnostic output if you do not run **cupsaddsmb** in verbose mode. Therefore, we strongly recommend to not use the default quiet mode. It will hide any problems from you that might occur.

18.10.13 cupsaddsmb with a Samba PDC

Can't get the standard **cupsaddsmb** command to run on a Samba PDC? Are you asked for the password credential all over again and again and the command just will not take off at all? Try one of these variations:

```
root# cupsaddsmb -U MIDEARTH\\root -v printername
```

```
root# cupsaddsmb -H SAURON -U MIDEARTH\\root -v printername
root# cupsaddsmb -H SAURON -U MIDEARTH\\root -h cups-server -v printername
```

(Note the two backslashes: the first one is required to “escape” the second one).

18.10.14 cupsaddsmb Flowchart

Figure 18.16 shows a chart about the procedures, commandflows and dataflows of the **cupsaddsmb** command. Note again: cupsaddsmb is not intended to, and does not work with, raw queues!

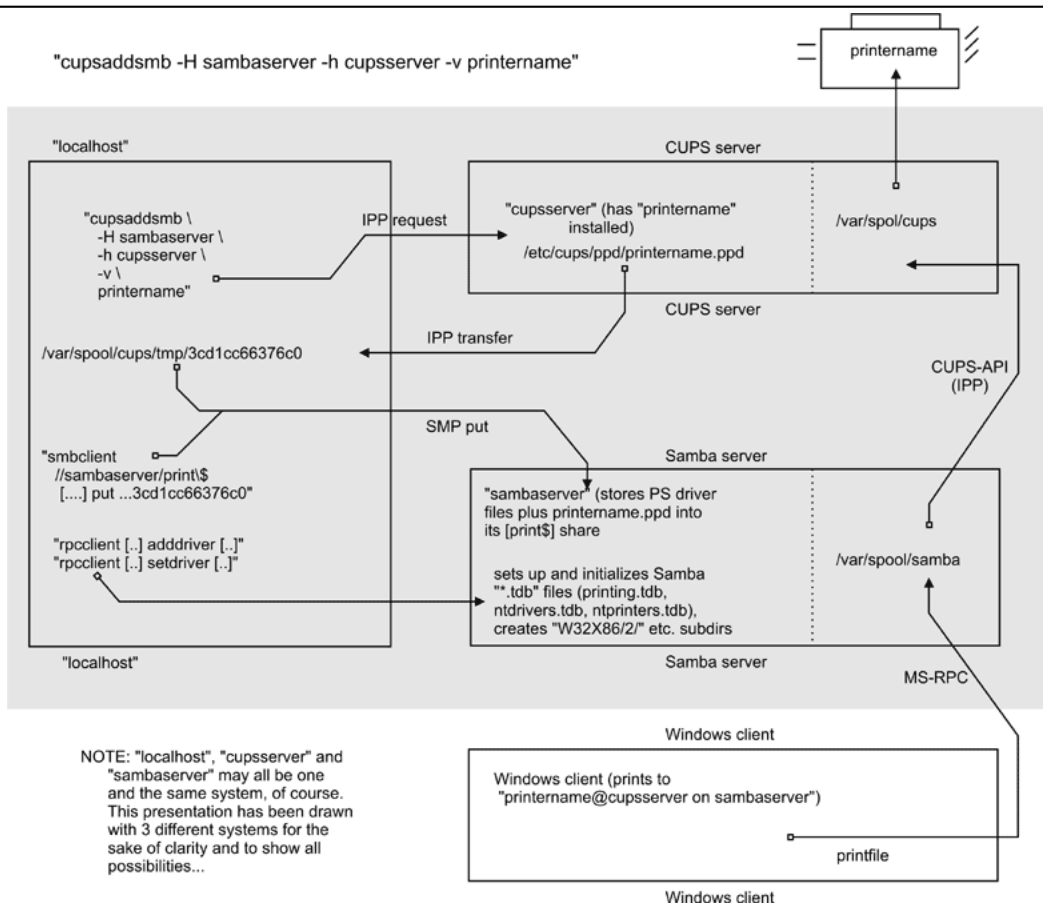


Figure 18.16. cupsaddsmb flowchart.

18.10.15 Installing the PostScript Driver on a Client

After **cupsaddsmb** is completed, your driver is prepared for the clients to use. Here are the steps you must perform to download and install it via Point'n'Print. From a Windows

client, browse to the CUPS/Samba server:

- Open the **Printers** share of Samba in Network Neighborhood.
- Right-click on the printer in question.
- From the opening context-menu select **Install...** or **Connect...** (depending on the Windows version you use).

After a few seconds, there should be a new printer in your client's *local Printers* folder. On Windows XP it will follow a naming convention of *PrinterName on SambaServer*. (In my current case it is "infotec-2105 on kde-bitshop"). If you want to test it and send your first job from an application like Winword, the new printer appears in a `\\SambaServer\PrinterName` entry in the drop-down list of available printers.

`cupsaddsmb` will only reliably work with CUPS version 1.1.15 or higher and Samba from 2.2.4. If it does not work, or if the automatic printer driver download to the clients does not succeed, you can still manually install the CUPS printer PPD on top of the Adobe PostScript driver on clients. Then point the client's printer queue to the Samba printer share for a UNC type of connection:

```
C:\> net use lpt1: \\sambaserver\printershare /user:ntadmin
```

should you desire to use the CUPS networked PostScript RIP functions. (Note that user "ntadmin" needs to be a valid Samba user with the required privileges to access the printershare.) This sets up the printer connection in the traditional *LanMan* way (not using MS-RPC).

18.10.16 Avoiding Critical PostScript Driver Settings on the Client

Printing works, but there are still problems. Most jobs print well, some do not print at all. Some jobs have problems with fonts, which do not look very good. Some jobs print fast and some are dead-slow. Many of these problems can be greatly reduced or even completely eliminated if you follow a few guidelines. Remember, if your print device is not PostScript-enabled, you are treating your Ghostscript installation on your CUPS host with the output your client driver settings produce. Treat it well:

- Avoid the PostScript Output Option: Optimize for Speed setting. Use the Optimize for Portability instead (Adobe PostScript driver).
- Don't use the Page Independence: NO setting. Instead, use Page Independence YES (CUPS PostScript Driver).
- Recommended is the True Type Font Downloading Option: Native True Type over Automatic and Outline; you should by all means avoid Bitmap (Adobe PostScript Driver).
- Choose True Type Font: Download as Softfont into Printer over the default Replace by Device Font (for exotic fonts, you may need to change it back to get a printout at all) (Adobe).

- Sometimes you can choose PostScript Language Level: In case of problems try 2 instead of 3 (the latest ESP Ghostscript package handles Level 3 PostScript very well) (Adobe).
- Say Yes to PostScript Error Handler (Adobe).

18.11 Installing PostScript Driver Files Manually Using `rpcclient`

Of course, you can run all the commands that are embedded into the `cupsaddsmb` convenience utility yourself, one by one, and hereby upload and prepare the driver files for future client downloads.

1. Prepare Samba (A CUPS print queue with the name of the printer should be there. We are providing the driver now).
2. Copy all files to `[print$]`.
3. Run `rpcclient adddriver` (for each client architecture you want to support).
4. Run `rpcclient setdriver`.

We are going to do this now. First, read the man page on `rpcclient` to get a first idea. Look at all the printing related subcommands. `enumprinters`, `enumdrivers`, `enumports`, `adddriver`, `setdriver` are among the most interesting ones. `rpcclient` implements an important part of the MS-RPC protocol. You can use it to query (and command) a Windows NT (or 200x/XP) PC, too. MS-RPC is used by Windows clients, among other things, to benefit from the Point'n'Print features. Samba can now mimic this as well.

18.11.1 A Check of the `rpcclient` man Page

First let's check the `rpcclient` man page. Here are two relevant passages:

`adddriver <arch> <config>` Execute an `AddPrinterDriver()` RPC to install the printer driver information on the server. The driver files should already exist in the directory returned by `getdriverdir`. Possible values for `arch` are the same as those for the `getdriverdir` command. The `config` parameter is defined as follows:

```
Long Printer Name:\
Driver File Name:\
Data File Name:\
Config File Name:\
Help File Name:\
Language Monitor Name:\
Default Data Type:\
Comma Separated list of Files
```

Any empty fields should be enter as the string `"NULL"`.

Samba does not need to support the concept of Print Monitors since these only apply to local printers whose driver can make use of a bi-directional link for communication. This

field should be “*NULL*”. On a remote NT print server, the Print Monitor for a driver must already be installed prior to adding the driver or else the RPC will fail.

setdriver <printername> <drivername> Execute a **SetPrinter()** command to update the printer driver associated with an installed printer. The printer driver must already be correctly installed on the print server.

See also the **enumprinters** and **enumdrivers** commands for obtaining a list of installed printers and drivers.

18.11.2 Understanding the `rpcclient` man Page

The *exact* format isn’t made too clear by the man page, since you have to deal with some parameters containing spaces. Here is a better description for it. We have line-broken the command and indicated the breaks with “\”. Usually you would type the command in one line without the linebreaks:

```
adddriver "Architecture" \
    "LongPrinterName:DriverFile:DataFile:ConfigFile:HelpFile:\
    LanguageMonitorFile:DataType:ListOfFiles,Comma-separated"
```

What the man pages denote as a simple <*config*> keyword, in reality consists of eight colon-separated fields. The last field may take multiple (in some very insane cases, even 20 different additional) files. This might sound confusing at first. What the man pages names the “*LongPrinterName*” in reality should be called the “*Driver Name*”. You can name it anything you want, as long as you use this name later in the **rpcclient ... setdriver** command. For practical reasons, many name the driver the same as the printer.

It isn’t simple at all. I hear you asking: “*How do I know which files are “Driver File”, “Data File”, “Config File”, “Help File” and “Language Monitor File” in each case?*” — For an answer, you may want to have a look at how a Windows NT box with a shared printer presents the files to us. Remember, that this whole procedure has to be developed by the Samba team by overhearing the traffic caused by Windows computers on the wire. We may as well turn to a Windows box now and access it from a UNIX workstation. We will query it with **rpcclient** to see what it tells us and try to understand the man page more clearly that we’ve read just now.

18.11.3 Producing an Example by Querying a Windows Box

We could run **rpcclient** with a **getdriver** or a **getprinter** subcommand (in level 3 verbosity) against it. Just sit down at a UNIX or Linux workstation with the Samba utilities installed, then type the following command:

```
root# rpcclient -U'user%secret' NT-SERVER -c 'getdriver printername 3'
```

From the result it should become clear which is which. Here is an example from my installation:


```

root# rpcclient -U'Danka%xxxx' W200xSERVER \
  -c'getdriver "DANKA InfoStream Virtual Printer" 3'
cmd = getdriver "DANKA InfoStream Virtual Printer" 3

[Windows NT x86]
Printer Driver Info 3:
  Version: [2]
  Driver Name: [DANKA InfoStream]
  Architecture: [Windows NT x86]
  Driver Path: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\PSCRIPT.DLL]
  Datafile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\INFOSTRM.PPD]
  Configfile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\PSCRPTUI.DLL]
  Helpfile: [C:\WINNT\System32\spool\DRIVERS\W32X86\2\PSCRIPT.HLP]

  Dependentsfiles: []
  Dependentsfiles: []
  Dependentsfiles: []
  Dependentsfiles: []
  Dependentsfiles: []
  Dependentsfiles: []
  Dependentsfiles: []

  Monitorname: []
  Defaultdatatype: []

```

Some printer drivers list additional files under the label *Dependentfiles* and these would go into the last field *ListOfFiles,Comma-separated*. For the CUPS PostScript drivers, we do not need any (nor would we for the Adobe PostScript driver), therefore, the field will get a “*NULL*” entry.

18.11.4 Requirements for adddriver and setdriver to Succeed

>From the man page (and from the quoted output of `cupsaddsmb` above) it becomes clear that you need to have certain conditions in order to make the manual uploading and initializing of the driver files succeed. The two `rpcclient` subcommands (`adddriver` and `setdriver`) need to encounter the following preconditions to complete successfully:

- You are connected as *printer admin* or root (this is *not* the “*Printer Operators*” group in NT, but the *printer admin* group as defined in the `[global]` section of `smb.conf`).
- Copy all required driver files to `\\SAMBA\print$\w32x86` and `\\SAMBA\print$\win40` as appropriate. They will end up in the “0” respective “2” subdirectories later. For now, *do not* put them there, they’ll be automatically used by the `adddriver` subcommand. (If you use `smbclient` to put the driver files into the share, note that you need to escape the “\$”: `smbclient //sambaserver/print/$ -U root.`)

- The user you're connecting as must be able to write to the `[print$]` share and create subdirectories.
- The printer you are going to setup for the Windows clients needs to be installed in CUPS already.
- The CUPS printer must be known to Samba, otherwise the `setdriver` subcommand fails with an `NT_STATUS_UNSUCCESSFUL` error. To check if the printer is known by Samba, you may use the `enumprinters` subcommand to `rpcclient`. A long-standing bug prevented a proper update of the printer list until every `smbd` process had received a `SIGHUP` or was restarted. Remember this in case you've created the CUPS printer just recently and encounter problems: try restarting Samba.

18.11.5 Manual Driver Installation in 15 Steps

We are going to install a printer driver now by manually executing all required commands. As this may seem a rather complicated process at first, we go through the procedure step by step, explaining every single action item as it comes up.

MANUAL DRIVER INSTALLATION

1. Install the printer on CUPS.

```
root# lpadmin -p mysmbtstprn -v socket://10.160.51.131:9100 -E \
        -P canonIR85.ppd
```

This installs a printer with the name `mysmbtstprn` to the CUPS system. The printer is accessed via a socket (a.k.a. JetDirect or Direct TCP/IP) connection. You need to be root for this step.

2. (Optional) Check if the printer is recognized by Samba.

```
root# rpcclient -Uroot%xxxx -c 'enumprinters' localhost \
| grep -C2 mysmbtstprn
flags: [0x800000]
name: [\\kde-bitshop\mysmbtstprn]
description: [\\kde-bitshop\mysmbtstprn,,mysmbtstprn]
comment: [mysmbtstprn]
```

This should show the printer in the list. If not, stop and restart the Samba daemon (`smbd`), or send a `HUP` signal:

```
root# kill -HUP `pidof smbd`
```

Check again. Troubleshoot and repeat until successful. Note the “*empty*” field between the two commas in the “*description*” line. The driver name would appear here if there was one already. You need to know root's Samba password (as set by the

`smbpasswd` command) for this step and most of the following steps. Alternately, you can authenticate as one of the users from the “*write list*” as defined in `smb.conf` for `[print$]`.

3. (Optional) Check if Samba knows a driver for the printer.

```
root# rpcclient -Uroot%xxxx -c 'getprinter mysmbtstprn 2' localhost \
    | grep driver
drivename: []
```

```
root# rpcclient -Uroot%xxxx -c 'getprinter mysmbtstprn 2' localhost \
    | grep -C4 driv
servername: [\\kde-bitshop]
printername: [\\kde-bitshop\mysmbtstprn]
sharename: [mysmbtstprn]
portname: [Samba Printer Port]
drivename: []
comment: [mysmbtstprn]
location: []
sepfile: []
printprocessor: [winprint]
```

```
root# rpcclient -U root%xxxx -c 'getdriver mysmbtstprn' localhost
result was WERR_UNKNOWN_PRINTER_DRIVER
```

None of the three commands shown above should show a driver. This step was done for the purpose of demonstrating this condition. An attempt to connect to the printer at this stage will prompt the message along the lines of: “*The server does not have the required printer driver installed.*”

4. Put all required driver files into Samba’s `[print$]`.

```
root# smbclient //localhost/print\$ -U 'root%xxxx' \
    -c 'cd W32X86; \
    put /etc/cups/ppd/mysmbtstprn.ppd mysmbtstprn.PPD; \
    put /usr/share/cups/drivers/cupsui.dll cupsui.dll; \
    put /usr/share/cups/drivers/cupsdrv.dll cupsdrv.dll; \
    put /usr/share/cups/drivers/cups.hlp cups.hlp'
```

(This command should be entered in one long single line. Line-breaks and the line-end indicated by “\” have been inserted for readability reasons.) This step is *required* for the next one to succeed. It makes the driver files physically present in the `[print$]` share. However, clients would still not be able to install them, because Samba does not yet treat them as driver files. A client asking for the driver would still be presented with a “*not installed here*” message.

5. Verify where the driver files are now.

```
root# ls -l /etc/samba/drivers/W32X86/
total 669
drwxr-sr-x   2 root   ntadmin   532 May 25 23:08 2
drwxr-sr-x   2 root   ntadmin   670 May 16 03:15 3
-rwxr--r--   1 root   ntadmin  14234 May 25 23:21 cups.hlp
-rwxr--r--   1 root   ntadmin 278380 May 25 23:21 cupsdrv.dll
-rwxr--r--   1 root   ntadmin 215848 May 25 23:21 cupsui.dll
-rwxr--r--   1 root   ntadmin 169458 May 25 23:21 mysmbtstprn.PPD
```

The driver files now are in the W32X86 architecture “root” of `[print$]`.

6. Tell Samba that these are driver files (adddriver).

```
root# rpcclient -Uroot%xxxx -c 'adddriver "Windows NT x86" \
  "mydrivername:cupsdrv.dll:mysmbtstprn.PPD: \
  cupsui.dll:cups.hlp:NULL:RAW:NULL" \
  localhost
Printer Driver mydrivername successfully installed.
```

You cannot repeat this step if it fails. It could fail even as a result of a simple typo. It will most likely have moved a part of the driver files into the “2” subdirectory. If this step fails, you need to go back to the fourth step and repeat it before you can try this one again. In this step, you need to choose a name for your driver. It is normally a good idea to use the same name as is used for the printer name; however, in big installations you may use this driver for a number of printers that obviously have different names, so the name of the driver is not fixed.

7. Verify where the driver files are now.

```
root# ls -l /etc/samba/drivers/W32X86/
total 1
drwxr-sr-x   2 root   ntadmin   532 May 25 23:22 2
drwxr-sr-x   2 root   ntadmin   670 May 16 03:15 3

root# ls -l /etc/samba/drivers/W32X86/2
total 5039
[... ]
-rwxr--r--   1 root   ntadmin  14234 May 25 23:21 cups.hlp
-rwxr--r--   1 root   ntadmin 278380 May 13 13:53 cupsdrv.dll
-rwxr--r--   1 root   ntadmin 215848 May 13 13:53 cupsui.dll
-rwxr--r--   1 root   ntadmin 169458 May 25 23:21 mysmbtstprn.PPD
```

Notice how step 6 also moved the driver files to the appropriate subdirectory. Compare this with the situation after step 5.

8. (Optional) Verify if Samba now recognizes the driver.

```

root# rpcclient -Uroot%xxxx -c 'enumdrivers 3' \
    localhost | grep -B2 -A5 mydrivername
Printer Driver Info 3:
Version: [2]
Driver Name: [mydrivername]
Architecture: [Windows NT x86]
Driver Path: [\\kde-bitshop\print$\W32X86\2\cupsdrv.dll]
Datafile: [\\kde-bitshop\print$\W32X86\2\mysmbtstprn.PPD]
Configfile: [\\kde-bitshop\print$\W32X86\2\cupsui.dll]
Helpfile: [\\kde-bitshop\print$\W32X86\2\cups.hlp]

```

Remember, this command greps for the name you chose for the driver in step 6. This command must succeed before you can proceed.

9. Tell Samba which printer should use these driver files (**setdriver**).

```

root# rpcclient -Uroot%xxxx -c 'setdriver mysmbtstprn mydrivername' \
    localhost
Successfully set mysmbtstprn to driver mydrivername

```

Since you can bind any printername (print queue) to any driver, this is a convenient way to setup many queues that use the same driver. You do not need to repeat all the previous steps for the **setdriver** command to succeed. The only preconditions are: **enumdrivers** must find the driver and **enumprinters** must find the printer.

10. (Optional) Verify if Samba has recognized this association.

```

root# rpcclient -Uroot%xxxx -c 'getprinter mysmbtstprn 2' localhost \
    | grep driver
drivername: [mydrivername]

root# rpcclient -Uroot%xxxx -c 'getprinter mysmbtstprn 2' localhost \
    | grep -C4 driv
servername: [\\kde-bitshop]
printername: [\\kde-bitshop\mysmbtstprn]
sharename: [mysmbtstprn]
portname: [Done]
drivername: [mydrivername]
comment: [mysmbtstprn]
location: []
sepfile: []
printprocessor: [winprint]

root# rpcclient -U root%xxxx -c 'getdriver mysmbtstprn' localhost

```

```
[Windows NT x86]
Printer Driver Info 3:
  Version: [2]
  Driver Name: [mydrivername]
  Architecture: [Windows NT x86]
  Driver Path: [\\kde-bitshop\print$\W32X86\2\cupsdrv.dll]
  Datafile: [\\kde-bitshop\print$\W32X86\2\mysmbtstprn.PPD]
  Configfile: [\\kde-bitshop\print$\W32X86\2\cupsui.dll]
  Helpfile: [\\kde-bitshop\print$\W32X86\2\cups.hlp]
  Monitorname: []
  Defaultdatatype: [RAW]
  Monitorname: []
  Defaultdatatype: [RAW]
```

```
root# rpcclient -Uroot%xxxx -c 'enumprinters' localhost \
| grep mysmbtstprn
name: [\\kde-bitshop\mysmbtstprn]
description: [\\kde-bitshop\mysmbtstprn,mydrivername,mysmbtstprn]
comment: [mysmbtstprn]
```

Compare these results with the ones from steps 2 and 3. Every one of these commands show the driver is installed. Even the **enumprinters** command now lists the driver on the “*description*” line.

11. **(Optional) Tickle the driver into a correct device mode.** You certainly know how to install the driver on the client. In case you are not particularly familiar with Windows, here is a short recipe: Browse the Network Neighborhood, go to the Samba server, and look for the shares. You should see all shared Samba printers. Double-click on the one in question. The driver should get installed and the network connection set up. An alternate way is to open the **Printers (and Faxes)** folder, right-click on the printer in question and select **Connect** or **Install**. As a result, a new printer should have appeared in your client’s local **Printers (and Faxes)** folder, named something like **printersharename on Sambahostname**. It is important that you execute this step as a Samba printer admin (as defined in `smb.conf`). Here is another method to do this on Windows XP. It uses a command line, which you may type into the “*DOS box*” (type root’s `smbpassword` when prompted):

```
C:\> runas /netonly /user:root "rundll32 printui.dll,PrintUIEntry \
/in /n \\sambaserver\mysmbtstprn"
```

Change any printer setting once (like changing **portrait** to **landscape**), click on **Apply**; change the setting back.

12. **Install the printer on a client (Point’n’Print).**

```
C:\> rundll32 printui.dll,PrintUIEntry /in /n \\sambaserver\mysmbtstprn
```

If it does not work it could be a permission problem with the `[print$]` share.

13. (Optional) **Print a test page.**

```
C:\> rundll32 printui.dll,PrintUIEntry /p /n "\\sambaserver\mysmbtstprn"
```

Then hit [TAB] five times, [ENTER] twice, [TAB] once and [ENTER] again and march to the printer.

14. (Recommended) **Study the test page.** Hmmm.... just kidding! By now you know everything about printer installations and you do not need to read a word. Just put it in a frame and bolt it to the wall with the heading "MY FIRST RPCCLIENT-INSTALLED PRINTER" — why not just throw it away!

15. (Obligatory) **Enjoy. Jump. Celebrate your success.**

```
root# echo "Cheeeeeriooooo! Success..." >> /var/log/samba/log.smbd
```

18.11.6 Troubleshooting Revisited

The `setdriver` command will fail, if in Samba's mind the queue is not already there. You had promising messages about the:

```
Printer Driver ABC successfully installed.
```

after the **adddriver** parts of the procedure? But you are also seeing a disappointing message like this one?

```
result was NT_STATUS_UNSUCCESSFUL
```

It is not good enough that you can see the queue in CUPS, using the `lpstat -p ir85wm` command. A bug in most recent versions of Samba prevents the proper update of the queuelist. The recognition of newly installed CUPS printers fails unless you restart Samba or send a HUP to all `smbd` processes. To verify if this is the reason why Samba does not execute the `setdriver` command successfully, check if Samba "sees" the printer:

```
root# rpcclient transmeta -N -U'root%xxxx' -c 'enumprinters 0'|grep ir85wm
  printername:[ir85wm]
```

An alternate command could be this:

```
root# rpcclient transmeta -N -U'root%secret' -c 'getprinter ir85wm'
  cmd = getprinter ir85wm
```

```
flags: [0x800000]
name: [\\transmeta\ir85wm]
description: [\\transmeta\ir85wm,ir85wm,DPD]
comment: [CUPS PostScript-Treiber for Windows NT/200x/XP]
```

By the way, you can use these commands, plus a few more, of course, to install drivers on remote Windows NT print servers too!

18.12 The Printing *.tdb Files

Some mystery is associated with the series of files with a tdb suffix appearing in every Samba installation. They are `connections.tdb`, `printing.tdb`, `share_info.tdb`, `ntdrivers.tdb`, `unexpected.tdb`, `brlock.tdb`, `locking.tdb`, `ntforms.tdb`, `messages.tdb`, `ntprinters.tdb`, `sessionid.tdb` and `secrets.tdb`. What is their purpose?

18.12.1 Trivial Database Files

A Windows NT (print) server keeps track of all information needed to serve its duty toward its clients by storing entries in the Windows registry. Client queries are answered by reading from the registry, Administrator or user configuration settings that are saved by writing into the registry. Samba and UNIX obviously do not have such a Registry. Samba instead keeps track of all client related information in a series of *.tdb files. (TDB = Trivial Data Base). These are often located in `/var/lib/samba/` or `/var/lock/samba/`. The printing related files are `ntprinters.tdb`, `printing.tdb`, `ntforms.tdb` and `ntdrivers.tdb`.

18.12.2 Binary Format

*.tdb files are not human readable. They are written in a binary format. “*Why not ASCII?*”, you may ask. “*After all, ASCII configuration files are a good and proven tradition on UNIX.*” The reason for this design decision by the Samba team is mainly performance. Samba needs to be fast; it runs a separate `smbd` process for each client connection, in some environments many thousands of them. Some of these `smbds` might need to write-access the same *.tdb file *at the same time*. The file format of Samba’s *.tdb files allows for this provision. Many `smbd` processes may write to the same *.tdb file at the same time. This wouldn’t be possible with pure ASCII files.

18.12.3 Losing *.tdb Files

It is very important that all *.tdb files remain consistent over all write and read accesses. However, it may happen that these files *do* get corrupted. (A `kill -9 'pidof smbd'` while a write access is in progress could do the damage as well as a power interruption, etc.). In cases of trouble, a deletion of the old printing-related *.tdb files may be the only option. After that you need to re-create all print-related setup or you have made a backup of the *.tdb files in time.

18.12.4 Using tdbbackup

Samba ships with a little utility that helps the root user of your system to backup your *.tdb files. If you run it with no argument, it prints a usage message:

```
root# tdbbackup
Usage: tdbbackup [options] <fname...>

Version:3.0a
  -h          this help message
  -s suffix   set the backup suffix
  -v          verify mode (restore if corrupt)
```

Here is how I backed up my printing.tdb file:

```
root# ls
.          browse.dat      locking.tdb      ntdrivers.tdb  printing.tdb
..         share_info.tdb  connections.tdb  messages.tdb   ntforms.tdb
printing.tdbkp  unexpected.tdb  brlock.tdb      gmon.out       namelist.debug
ntprinters.tdb  sessionid.tdb
```

```
root# tdbbackup -s .bak printing.tdb
printing.tdb : 135 records
```

```
root# ls -l printing.tdb*
-rw-----  1 root    root      40960 May  2 03:44 printing.tdb
-rw-----  1 root    root      40960 May  2 03:44 printing.tdb.bak
```

18.13 CUPS Print Drivers from Linuxprinting.org

CUPS ships with good support for HP LaserJet-type printers. You can install the generic driver as follows:

```
root# lpadmin -p laserjet4plus -v parallel:/dev/lp0 -E -m laserjet.ppd
```

The `-m` switch will retrieve the `laserjet.ppd` from the standard repository for not-yet-installed-PPDs, which CUPS typically stores in `/usr/share/cups/model`. Alternately, you may use `-P /path/to/your.ppd`.

The generic `laserjet.ppd`, however, does not support every special option for every LaserJet-compatible model. It constitutes a sort of “*least common denominator*” of all the models. If for some reason you must pay for the commercially available ESP Print Pro drivers, your

first move should be to consult the database on http://www.linuxprinting.org/printer_list.cgi. Linuxprinting.org has excellent recommendations about which driver is best used for each printer. Its database is kept current by the tireless work of Till Kampeter from MandrakeSoft, who is also the principal author of the **foomatic-rip** utility.

NOTE



The former **cupsonomatic** concept is now being replaced by the new successor, a much more powerful **foomatic-rip**. **cupsonomatic** is no longer maintained. Here is the new URL to the Foomatic-3.0 database: http://www.linuxprinting.org/driver_list.cgi. If you upgrade to **foomatic-rip**, remember to also upgrade to the new-style PPDs for your Foomatic-driven printers. **foomatic-rip** will not work with PPDs generated for the old **cupsonomatic**. The new-style PPDs are 100% compliant to the Adobe PPD specification. They are also intended to be used by Samba and the **cupssaddsmb** utility, to provide the driver files for the Windows clients!

18.13.1 foomatic-rip and Foomatic Explained

Nowadays, most Linux distributions rely on the utilities of Linuxprinting.org to create their printing-related software (which, by the way, works on all UNIXes and on Mac OS X or Darwin, too). It is not known as well as it should be, that it also has a very end-user-friendly interface that allows for an easy update of drivers and PPDs for all supported models, all spoolers, all operating systems, and all package formats (because there is none). Its history goes back a few years.

Recently, Foomatic has achieved the astonishing milestone of 1000 listed¹² printer models. Linuxprinting.org keeps all the important facts about printer drivers, supported models and which options are available for the various driver/printer combinations in its Foomatic¹³ database. Currently there are 245 drivers¹⁴ in the database. Many drivers support various models, and many models may be driven by different drivers — its your choice!

18.13.1.1 690 “Perfect” Printers

At present, there are 690 devices dubbed as working perfectly, 181 mostly, 96 partially, and 46 are paperweights. Keeping in mind that most of these are non-PostScript models (PostScript printers are automatically supported by CUPS to perfection, by using their own manufacturer-provided Windows-PPD), and that a multifunctional device never qualifies as working perfectly if it does not also scan and copy and fax under GNU/Linux — then this is a truly astonishing achievement! Three years ago the number was not more than 500, and Linux or UNIX printing at the time wasn’t anywhere near the quality it is today.

¹²http://www.linuxprinting.org/printer_list.cgi?make=Anyone

¹³<http://www.linuxprinting.org/foomatic.html>

¹⁴http://www.linuxprinting.org/driver_list.cgi

18.13.1.2 How the Printing HOWTO Started It All

A few years ago Grant Taylor¹⁵ started it all. The roots of today's Linuxprinting.org are in the first Linux Printing HOWTO¹⁶ that he authored. As a side-project to this document, which served many Linux users and admins to guide their first steps in this complicated and delicate setup (to a scientist, printing is “*applying a structured deposition of distinct patterns of ink or toner particles on paper substrates*”), he started to build in a little Postgres database with information about the hardware and driver zoo that made up Linux printing of the time. This database became the core component of today's Foomatic collection of tools and data. In the meantime, it has moved to an XML representation of the data.

18.13.1.3 Foomatic's Strange Name

“*Why the funny name?*” you ask. When it really took off, around spring 2000, CUPS was far less popular than today, and most systems used LPD, LPRng or even PDQ to print. CUPS shipped with a few generic drivers (good for a few hundred different printer models). These didn't support many device-specific options. CUPS also shipped with its own built-in rasterization filter (*pstoraster*, derived from Ghostscript). On the other hand, CUPS provided brilliant support for *controlling* all printer options through standardized and well-defined PPD files (PostScript Printers Description files). Plus, CUPS was designed to be easily extensible.

Taylor already had in his database a respectable compilation of facts about many more printers and the Ghostscript “*drivers*” they run with. His idea, to generate PPDs from the database information and use them to make standard Ghostscript filters work within CUPS, proved to work very well. It also killed several birds with one stone:

- It made all current and future Ghostscript filter developments available for CUPS.
- It made available a lot of additional printer models to CUPS users (because often the traditional Ghostscript way of printing was the only one available).
- It gave all the advanced CUPS options (Web interface, GUI driver configurations) to users wanting (or needing) to use Ghostscript filters.

18.13.1.4 cupsomatic, pdqomatic, lpdomatic, directomatic

CUPS worked through a quickly-hacked up filter script named cupsomatic.¹⁷ cupsomatic ran the printfile through Ghostscript, constructing automatically the rather complicated command line needed. It just needed to be copied into the CUPS system to make it work. To configure the way cupsomatic controls the Ghostscript rendering process, it needs a CUPS-PPD. This PPD is generated directly from the contents of the database. For CUPS and the respective printer/filter combo, another Perl script named CUPS-O-Matic did the PPD generation. After that was working, Taylor implemented within a few days a similar

¹⁵<http://www2.picante.com:81/~gtaylor/>

¹⁶<http://www.linuxprinting.org/foomatic2.9/howto/>

¹⁷<http://www.linuxprinting.org/download.cgi?filename=cupsomatic&show=0>

thing for two other spoolers. Names chosen for the config-generator scripts were PDQ-O-Matic¹⁸ (for PDQ) and LPD-O-Matic¹⁹ (for — you guessed it — LPD); the configuration here didn't use PPDs but other spooler-specific files.

From late summer of that year, Till Kamppeter²⁰ started to put work into the database. Kamppeter had been newly employed by MandrakeSoft²¹ to convert its printing system over to CUPS, after they had seen his FLTK²²-based XPP²³ (a GUI frontend to the CUPS lp-command). He added a huge amount of new information and new printers. He also developed the support for other spoolers, like PPR²⁴ (via ppromatic), GNUlpr²⁵ and LPRng²⁶ (both via an extended lpdomatic) and spoolerless printing (directomatic²⁷).

So, to answer your question: “*Foomatic*” is the general name for all the overlapping code and data behind the “**omatic*” scripts. Foomatic, up to versions 2.0.x, required (ugly) Perl data structures attached to Linuxprinting.org PPDs for CUPS. It had a different “**omatic*” script for every spooler, as well as different printer configuration files.

18.13.1.5 The *Grand Unification Achieved*

This has all changed in Foomatic versions 2.9 (beta) and released as “*stable*” 3.0. It has now achieved the convergence of all **omatic* scripts and is called the *foomatic-rip*.²⁸ This single script is the unification of the previously different spooler-specific **omatic* scripts. *foomatic-rip* is used by all the different spoolers alike and because it can read PPDs (both the original PostScript printer PPDs and the Linuxprinting.org-generated ones), all of a sudden all supported spoolers can have the power of PPDs at their disposal. Users only need to plug *foomatic-rip* into their system. For users there is improved media type and source support — paper sizes and trays are easier to configure.

Also, the New Generation of Linuxprinting.org PPDs no longer contains Perl data structures. If you are a distro maintainer and have used the previous version of Foomatic, you may want to give the new one a spin, but remember to generate a new-version set of PPDs via the new *foomatic-db-engine*!²⁹ Individual users just need to generate a single new PPD specific to their model by following the steps³⁰ outlined in the Foomatic tutorial or in this chapter. This new development is truly amazing.

foomatic-rip is a very clever wrapper around the need to run Ghostscript with a different syntax, options, device selections, and/or filters for each different printer or spooler. At the same time it can read the PPD associated with a print queue and modify the print job according to the user selections. Together with this comes the 100% compliance of the new

¹⁸<http://www.linuxprinting.org/download.cgi?filename=lpdomatic&show=0>

¹⁹<http://www.linuxprinting.org/download.cgi?filename=lpdomatic&show=0>

²⁰<http://www.linuxprinting.org/till/>

²¹<http://www.mandrakesoft.com/>

²²<http://www.fltk.org/>

²³<http://cups.sourceforge.net/xpp/>

²⁴<http://ppr.sourceforge.net/>

²⁵<http://sourceforge.net/projects/lpr/>

²⁶<http://www.lprng.org/>

²⁷<http://www.linuxprinting.org/download.cgi?filename=directomatic&show=0>

²⁸<http://www.linuxprinting.org/foomatic2.9/download.cgi?filename=foomatic-rip&show=0>

²⁹<http://www.linuxprinting.org/download/foomatic/foomatic-db-engine-3.0.0beta1.tar.gz>

³⁰<http://www.linuxprinting.org/kpfeifle/LinuxKongress2002/Tutorial/II.Foomatic-User/II.tutorial-handout-foomatic-user.html>

Foomatic PPDs with the Adobe spec. Some innovative features of the Foomatic concept may surprise users. It will support custom paper sizes for many printers and will support printing on media drawn from different paper trays within the same job (in both cases, even where there is no support for this from Windows-based vendor printer drivers).

18.13.1.6 Driver Development Outside

Most driver development itself does not happen within Linuxprinting.org. Drivers are written by independent maintainers. Linuxprinting.org just pools all the information and stores it in its database. In addition, it also provides the Foomatic glue to integrate the many drivers into any modern (or legacy) printing system known to the world.

Speaking of the different driver development groups, most of the work is currently done in three projects. These are:

- Omni³¹ — a free software project by IBM that tries to convert their printer driver knowledge from good-ol' OS/2 times into a modern, modular, universal driver architecture for Linux/UNIX (still beta). This currently supports 437 models.
- HPIJS³² — a free software project by HP to provide the support for their own range of models (very mature, printing in most cases is perfect and provides true photo quality). This currently supports 369 models.
- Gimp-Print³³ — a free software effort, started by Michael Sweet (also lead developer for CUPS), now directed by Robert Krawitz, which has achieved an amazing level of photo print quality (many Epson users swear that its quality is better than the vendor drivers provided by Epson for the Microsoft platforms). This currently supports 522 models.

18.13.1.7 Forums, Downloads, Tutorials, Howtos — also for Mac OS X and Commercial UNIX

Linuxprinting.org today is the one-stop shop to download printer drivers. Look for printer information and tutorials³⁴ or solve printing problems in its popular forums.³⁵ This forum it's not just for GNU/Linux users, but admins of commercial UNIX systems³⁶ are also going there, and the relatively new Mac OS X forum³⁷ has turned out to be one of the most frequented forums after only a few weeks.

Linuxprinting.org and the Foomatic driver wrappers around Ghostscript are now a standard toolchain for printing on all the important distros. Most of them also have CUPS underneath. While in recent years most printer data had been added by Kampeter (who works at Mandrake), many additional contributions came from engineers with SuSE, RedHat,

³¹<http://www-124.ibm.com/developerworks/oss/linux/projects/omni/>

³²<http://hpinkjet.sf.net/>

³³<http://gimp-print.sf.net/>

³⁴<http://www.linuxprinting.org//kpfeifle/LinuxKongress2002/Tutorial/>

³⁵<http://www.linuxprinting.org/newsportal/>

³⁶<http://www.linuxprinting.org/macosex/>

³⁷<http://www.linuxprinting.org/newsportal/thread.php3?name=linuxprinting.macosex.general>

Connectiva, Debian, and others. Vendor-neutrality is an important goal of the Foomatic project.

NOTE



Till Kämpeter from MandrakeSoft is doing an excellent job in his spare time to maintain [Linuxprinting.org](http://linuxprinting.org) and Foomatic. So if you use it often, please send him a note showing your appreciation.

18.13.1.8 Foomatic Database-Generated PPDs

The Foomatic database is an amazing piece of ingenuity in itself. Not only does it keep the printer and driver information, but it is organized in a way that it can generate PPD files on the fly from its internal XML-based datasets. While these PPDs are modelled to the Adobe specification of PostScript Printer Descriptions (PPDs), the [Linuxprinting.org](http://linuxprinting.org)/Foomatic-PPDs do not normally drive PostScript printers. They are used to describe all the bells and whistles you could ring or blow on an Epson Stylus inkjet, or a HP Photosmart, or what-have-you. The main trick is one little additional line, not envisaged by the PPD specification, starting with the **cupsFilter* keyword. It tells the CUPS daemon how to proceed with the PostScript print file (old-style Foomatic-PPDs named the cupsomatic filter script, while the new-style PPDs are now call foomatic-rip). This filter script calls Ghostscript on the host system (the recommended variant is ESP Ghostscript) to do the rendering work. foomatic-rip knows which filter or internal device setting it should ask from Ghostscript to convert the PostScript printjob into a raster format ready for the target device. This usage of PPDs to describe the options of non-PS printers was the invention of the CUPS developers. The rest is easy. GUI tools (like KDE's marvelous kprinter,³⁸ or the GNOME gtklp,³⁹ xpp and the CUPS Web interface) read the PPD as well and use this information to present the available settings to the user as an intuitive menu selection.

18.13.2 foomatic-rip and Foomatic-PPD Download and Installation

Here are the steps to install a foomatic-rip driven LaserJet 4 Plus-compatible printer in CUPS (note that recent distributions of SuSE, UnitedLinux and Mandrake may ship with a complete package of Foomatic-PPDs plus the **foomatic-rip** utility. Going directly to [Linuxprinting.org](http://linuxprinting.org) ensures that you get the latest driver/PPD files):

- Open your browser at the [Linuxprinting.org](http://linuxprinting.org) printer listpage.⁴⁰
- Check the complete list of printers in the database.⁴¹
- Select your model and click on the link.

³⁸<http://printing.kde.org/overview/kprinter.phtml>

³⁹<http://gtkpl.sourceforge.net/>

⁴⁰http://www.linuxprinting.org/printer_list.cgi

⁴¹http://www.linuxprinting.org/printer_list.cgi?make=Anyone

- You'll arrive at a page listing all drivers working with this model (for all printers, there will always be *one* recommended driver. Try this one first).
- In our case (HP LaserJet 4 Plus), we'll arrive at the default driver for the HP-LaserJet 4 Plus.⁴²
- The recommended driver is `ljet4`.
- Several links are provided here. You should visit them all if you are not familiar with the Linuxprinting.org database.
- There is a link to the database page for the `ljet4`.⁴³ On the driver's page, you'll find important and detailed information about how to use that driver within the various available spoolers.
- Another link may lead you to the homepage of the driver author or the driver.
- Important links are the ones that provide hints with setup instructions for CUPS, PDQ⁴⁴, LPD, LPRng and GNUlpr⁴⁵) as well as PPR⁴⁶ or “*spooler-less*” printing.⁴⁷
- You can view the PPD in your browser through this link: http://www.linuxprinting.org/ppd-o-matic.cgi?driver=ljet4&printer=HP-LaserJet_4_Plus&show=1
- Most importantly, you can also generate and download the PPD.⁴⁸
- The PPD contains all the information needed to use our model and the driver; once installed, this works transparently for the user. Later you'll only need to choose resolution, paper size, and so on from the Web-based menu, or from the print dialog GUI, or from the command line.
- If you ended up on the drivers page⁴⁹ you can choose to use the “*PPD-O-Matic*” online PPD generator program.
- Select the exact model and check either **Download** or **Display PPD file** and click **Generate PPD file**.
- If you save the PPD file from the browser view, please do not use cut and paste (since it could possibly damage line endings and tabs, which makes the PPD likely to fail its duty), but use **Save as...** in your browser's menu. (It is best to use the **Download** option directly from the Web page).
- Another interesting part on each driver page is the **Show execution details** button. If you select your printer model and click on that button, a complete Ghostscript command line will be displayed, enumerating all options available for that combination of driver and printer model. This is a great way to “*learn Ghostscript by doing*”. It is also an excellent cheat sheet for all experienced users who need to re-construct a good command line for that damn printing script, but can't remember the exact syntax.

⁴²http://www.linuxprinting.org/show_printer.cgi?recnum=HP-LaserJet_4_Plus

⁴³http://www.linuxprinting.org/show_driver.cgi?driver=ljet4

⁴⁴<http://www.linuxprinting.org/pdq-doc.html>

⁴⁵<http://www.linuxprinting.org/lpd-doc.html>

⁴⁶<http://www.linuxprinting.org/ppr-doc.html>

⁴⁷<http://www.linuxprinting.org/direct-doc.html>

⁴⁸http://www.linuxprinting.org/ppd-o-matic.cgi?driver=ljet4&printer=HP-LaserJet_4_Plus&show=0

⁴⁹http://www.linuxprinting.org/show_driver.cgi?driver=ljet4

- Some time during your visit to [Linuxprinting.org](http://linuxprinting.org), save the PPD to a suitable place on your harddisk, say `/path/to/my-printer.ppd` (if you prefer to install your printers with the help of the CUPS Web interface, save the PPD to the `/usr/share/cups/model/` path and restart `cupsd`).
- Then install the printer with a suitable command line, like this:

```
root# lpadmin -p laserjet4plus -v parallel:/dev/lp0 -E \
-P path/to/my-printer.ppd
```

- For all the new-style “*Foomatic-PPDs*” from [Linuxprinting.org](http://linuxprinting.org), you also need a special CUPS filter named `foomatic-rip`.
- The `foomatic-rip` Perlscript itself also makes some interesting reading⁵⁰ because it is well documented by Kampeter’s inline comments (even non-Perl hackers will learn quite a bit about printing by reading it).
- Save `foomatic-rip` either directly in `/usr/lib/cups/filter/foomatic-rip` or somewhere in your `$PATH` (and remember to make it world-executable). Again, do not save by copy and paste but use the appropriate link or the **Save as...** menu item in your browser.
- If you save `foomatic-rip` in your `$PATH`, create a symlink:

```
root# cd /usr/lib/cups/filter/ ; ln -s 'which foomatic-rip'
```

CUPS will discover this new available filter at startup after restarting `cupsd`.

Once you print to a print queue set up with the `Foomatic-PPD`, CUPS will insert the appropriate commands and comments into the resulting PostScript jobfile. `foomatic-rip` is able to read and act upon these and uses some specially encoded `Foomatic` comments embedded in the jobfile. These in turn are used to construct (transparently for you, the user) the complicated Ghostscript command line telling the printer driver exactly how the resulting raster data should look and which printer commands to embed into the data stream. You need:

- A “*foomatic+something*” PPD — but this is not enough to print with CUPS (it is only *one* important component).
- The *foomatic-rip* filter script (Perl) in `/usr/lib/cups/filters/`.
- Perl to make `foomatic-rip` run.
- Ghostscript (because it is doing the main work, controlled by the PPD/`foomatic-rip` combo) to produce the raster data fit for your printer model’s consumption.
- Ghostscript *must* (depending on the driver/model) contain support for a certain device representing the selected driver for your model (as shown by `gs -h`).

⁵⁰<http://www.linuxprinting.org/foomatic2.9/download.cgi?filename=foomatic-rip&show=1>

- foomatic-rip needs a new version of PPDs (PPD versions produced for cupsomatic do not work with foomatic-rip).

18.14 Page Accounting with CUPS

Often there are questions regarding print quotas where Samba users (that is, Windows clients) should not be able to print beyond a certain number of pages or data volume per day, week or month. This feature is dependent on the real print subsystem you're using. Samba's part is always to receive the job files from the clients (filtered *or* unfiltered) and hand it over to this printing subsystem.

Of course one could hack things with one's own scripts. But then there is CUPS. CUPS supports quotas that can be based on the size of jobs or on the number of pages or both, and span any time period you want.

18.14.1 Setting Up Quotas

This is an example command of how root would set a print quota in CUPS, assuming an existing printer named “*quotaprinter*”:

```
root# lpadmin -p quotaprinter -o job-quota-period=604800 \  
-o job-k-limit=1024 -o job-page-limit=100
```

This would limit every single user to print 100 pages or 1024 KB of data (whichever comes first) within the last 604,800 seconds (= 1 week).

18.14.2 Correct and Incorrect Accounting

For CUPS to count correctly, the printfile needs to pass the CUPS pstops filter, otherwise it uses a dummy count of “*one*”. Some print files do not pass it (e.g., image files) but then those are mostly one- page jobs anyway. This also means that proprietary drivers for the target printer running on the client computers and CUPS/Samba, which then spool these files as “*raw*” (i.e., leaving them untouched, not filtering them), will be counted as one-pagers too!

You need to send PostScript from the clients (i.e., run a PostScript driver there) to have the chance to get accounting done. If the printer is a non-PostScript model, you need to let CUPS do the job to convert the file to a print-ready format for the target printer. This is currently working for about a thousand different printer models. Linuxprinting has a driver list.⁵¹

18.14.3 Adobe and CUPS PostScript Drivers for Windows Clients

Before CUPS 1.1.16, your only option was to use the Adobe PostScript Driver on the Windows clients. The output of this driver was not always passed through the **psstops** filter

⁵¹http://www.linuxprinting.org/prINTER_list.cgi

on the CUPS/Samba side, and therefore was not counted correctly (the reason is that it often, depending on the PPD being used, wrote a PJI-header in front of the real PostScript which caused CUPS to skip **pstops** and go directly to the **pstoraster** stage).

From CUPS 1.1.16 onward, you can use the CUPS PostScript Driver for Windows NT/200x/XP clients (which is tagged in the download area of <http://www.cups.org/> as the `cups-samba-1.1.16.tar.gz` package). It does *not* work for Windows 9x/ME clients, but it guarantees:

- To not write a PJI-header.
- To still read and support all PJI-options named in the driver PPD with its own means.
- That the file will pass through the **pstops** filter on the CUPS/Samba server.
- To page-count correctly the print file.

You can read more about the setup of this combination in the man page for **cupsaddsmb** (which is only present with CUPS installed, and only current from CUPS 1.1.16).

18.14.4 The page_log File Syntax

These are the items CUPS logs in the `page_log` for every page of a job:

- Printer name
- User name
- Job ID
- Time of printing
- The page number
- The number of copies
- A billing information string (optional)
- The host that sent the job (included since version 1.1.19)

Here is an extract of my CUPS server's `page_log` file to illustrate the format and included items:

```
tec_IS2027 kurt 401 [22/Apr/2003:10:28:43 +0100] 1 3 #marketing 10.160.50.13
tec_IS2027 kurt 401 [22/Apr/2003:10:28:43 +0100] 2 3 #marketing 10.160.50.13
tec_IS2027 kurt 401 [22/Apr/2003:10:28:43 +0100] 3 3 #marketing 10.160.50.13
tec_IS2027 kurt 401 [22/Apr/2003:10:28:43 +0100] 4 3 #marketing 10.160.50.13
Dig9110 boss 402 [22/Apr/2003:10:33:22 +0100] 1 440 finance-dep 10.160.51.33
```

This was job ID `401`, printed on `tec_IS2027` by user `kurt`, a 64-page job printed in three copies and billed to `#marketing`, sent from IP address 10.160.50.13. The next job had ID `402`, was sent by user `boss` from IP address 10.160.51.33, printed from one page 440 copies and is set to be billed to `finance-dep`.

18.14.5 Possible Shortcomings

What flaws or shortcomings are there with this quota system?

- The ones named above (wrongly logged job in case of printer hardware failure, and so on).
- In reality, CUPS counts the job pages that are being processed in *software* (that is, going through the RIP) rather than the physical sheets successfully leaving the printing device. Thus if there is a jam while printing the fifth sheet out of a thousand and the job is aborted by the printer, the page count will still show the figure of a thousand for that job.
- All quotas are the same for all users (no flexibility to give the boss a higher quota than the clerk) and no support for groups.
- No means to read out the current balance or the “*used-up*” number of current quota.
- A user having used up 99 sheets of a 100 quota will still be able to send and print a thousand sheet job.
- A user being denied a job because of a filled-up quota does not get a meaningful error message from CUPS other than “*client-error-not-possible*”.

18.14.6 Future Developments

This is the best system currently available, and there are huge improvements under development for CUPS 1.2:

- Page counting will go into the backends (these talk directly to the printer and will increase the count in sync with the actual printing process; thus, a jam at the fifth sheet will lead to a stop in the counting).
- Quotas will be handled more flexibly.
- Probably there will be support for users to inquire about their accounts in advance.
- Probably there will be support for some other tools around this topic.

18.15 Additional Material

A printer queue with *no* PPD associated to it is a “*raw*” printer and all files will go directly there as received by the spooler. The exceptions are file types *application/octet-stream* that need passthrough feature enabled. “*Raw*” queues do not do any filtering at all, they hand the file directly to the CUPS backend. This backend is responsible for sending the data to the device (as in the “*device URI*” notation: *lpd://*, *socket://*, *smb://*, *ipp://*, *http://*, *parallel:/*, *serial:/*, *usb:/*, and so on).

cupsonic/*Foomatic* are *not* native CUPS drivers and they do not ship with CUPS. They are a third party add-on developed at [Linuxprinting.org](http://linuxprinting.org). As such, they are a brilliant hack to make all models (driven by Ghostscript drivers/filters in traditional spoolers) also work via CUPS, with the same (good or bad!) quality as in these other spoolers. *cupsonic* is only

a vehicle to execute a Ghostscript commandline at that stage in the CUPS filtering chain, where normally the native CUPS *pstoraster* filter would kick in. cupsomatic bypasses pstoraster, kidnaps the printfile from CUPS away and redirects it to go through Ghostscript. CUPS accepts this, because the associated cupsomatic/foomatic-PPD specifies:

```
*cupsFilter: "application/vnd.cups-postscript 0 cupsomatic"
```

This line persuades CUPS to hand the file to cupsomatic, once it has successfully converted it to the MIME type *application/vnd.cups-postscript*. This conversion will not happen for Jobs arriving from Windows that are auto-typed *application/octet-stream*, with the according changes in */etc/cups/mime.types* in place.

CUPS is widely configurable and flexible, even regarding its filtering mechanism. Another workaround in some situations would be to have in */etc/cups/mime.types* entries as follows:

```
application/postscript          application/vnd.cups-raw  0  -
application/vnd.cups-postscript application/vnd.cups-raw  0  -
```

This would prevent all PostScript files from being filtered (rather, they will through the virtual *nullfilter* denoted with “-”). This could only be useful for PS printers. If you want to print PS code on non-PS printers (provided they support ASCII text printing), an entry as follows could be useful:

```
*/*          application/vnd.cups-raw  0  -
```

and would effectively send *all* files to the backend without further processing.

You could have the following entry:

```
application/vnd.cups-postscript application/vnd.cups-raw 0 \
  my_PJL_stripping_filter
```

You will need to write a *my_PJL_stripping_filter* (which could be a shell script) that parses the PostScript and removes the unwanted PJL. This needs to conform to CUPS filter design (mainly, receive and pass the parameters printername, job-id, username, jobtitle, copies, print options and possibly the filename). It is installed as world executable into */usr/lib/cups/filters/* and is called by CUPS if it encounters a MIME type *application/vnd.cups-postscript*.

CUPS can handle *-o job-hold-until=indefinite*. This keeps the job in the queue on hold. It will only be printed upon manual release by the printer operator. This is a requirement in many central reproduction departments, where a few operators manage the jobs of hundreds of users on some big machine, where no user is allowed to have direct access (such

as when the operators often need to load the proper paper type before running the 10,000 page job requested by marketing for the mailing, and so on).

18.16 Auto-Deletion or Preservation of CUPS Spool Files

Samba print files pass through two spool directories. One is the incoming directory managed by Samba, (set in the *path* = /var/spool/samba directive in the *[printers]* section of *smb.conf*). The other is the spool directory of your UNIX print subsystem. For CUPS it is normally /var/spool/cups/, as set by the *cupsd.conf* directive *RequestRoot* /var/spool/cups/.

18.16.1 CUPS Configuration Settings Explained

Some important parameter settings in the CUPS configuration file *cupsd.conf* are:

PreserveJobHistory Yes — This keeps some details of jobs in *cupsd*'s mind (well it keeps the *c12345*, *c12346*, and so on, files in the CUPS spool directory, which do a similar job as the old-fashioned BSD-LPD control files). This is set to “*Yes*” as a default.

PreserveJobFiles Yes — This keeps the job files themselves in *cupsd*'s mind (it keeps the *d12345*, *d12346* etc. files in the CUPS spool directory). This is set to “*No*” as the CUPS default.

“**MaxJobs 500**” — This directive controls the maximum number of jobs that are kept in memory. Once the number of jobs reaches the limit, the oldest completed job is automatically purged from the system to make room for the new one. If all of the known jobs are still pending or active, then the new job will be rejected. Setting the maximum to 0 disables this functionality. The default setting is 0.

(There are also additional settings for *MaxJobsPerUser* and *MaxJobsPerPrinter...*)

18.16.2 Pre-Conditions

For everything to work as announced, you need to have three things:

- A Samba-*smbd* that is compiled against *libcups* (check on Linux by running `ldd 'which smbd'`).
- A Samba-*smb.conf* setting of *printing* = cups.
- Another Samba-*smb.conf* setting of *printcap* = cups.

NOTE



In this case, all other manually set printing-related commands (like *print command*, *lpq command*, *lprm command*, *lppause command* or *lpresume command*) are ignored and they should normally have no influence whatsoever on your printing.

18.16.3 Manual Configuration

If you want to do things manually, replace the *printing = cups* by *printing = bsd*. Then your manually set commands may work (I haven't tested this), and a *print command = lp -d %P %s; rm %s* may do what you need.

18.17 Printing from CUPS to Windows Attached Printers

>From time to time the question arises, how can you print *to* a Windows attached printer *from* Samba? Normally the local connection from Windows host to printer would be done by USB or parallel cable, but this does not matter to Samba. From here only an SMB connection needs to be opened to the Windows host. Of course, this printer must be shared first. As you have learned by now, CUPS uses *backends* to talk to printers and other servers. To talk to Windows shared printers, you need to use the *smb* (surprise, surprise!) backend. Check if this is in the CUPS backend directory. This usually resides in */usr/lib/cups/backend/*. You need to find an *smb* file there. It should be a symlink to *smbpool* and the file must exist and be executable:

```
root# ls -l /usr/lib/cups/backend/
total 253
drwxr-xr-x  3 root  root    720 Apr 30 19:04 .
drwxr-xr-x  6 root  root   125 Dec 19 17:13 ..
-rwxr-xr-x  1 root  root 10692 Feb 16 21:29 canon
-rwxr-xr-x  1 root  root 10692 Feb 16 21:29 epson
lrwxrwxrwx  1 root  root     3 Apr 17 22:50 http -> ipp
-rwxr-xr-x  1 root  root 17316 Apr 17 22:50 ipp
-rwxr-xr-x  1 root  root 15420 Apr 20 17:01 lpd
-rwxr-xr-x  1 root  root  8656 Apr 20 17:01 parallel
-rwxr-xr-x  1 root  root  2162 Mar 31 23:15 pdfdistiller
lrwxrwxrwx  1 root  root    25 Apr 30 19:04 ptal -> /usr/sbin/ptal-cups
-rwxr-xr-x  1 root  root  6284 Apr 20 17:01 scsi
lrwxrwxrwx  1 root  root    17 Apr  2 03:11 smb -> /usr/bin/smbpool
-rwxr-xr-x  1 root  root  7912 Apr 20 17:01 socket
-rwxr-xr-x  1 root  root  9012 Apr 20 17:01 usb

root# ls -l `which smbpool`
```

```
-rwxr-xr-x  1 root  root  563245 Dec 28 14:49 /usr/bin/smbpool
```

If this symlink does not exist, create it:

```
root# ln -s 'which smbpool' /usr/lib/cups/backend/smb
```

smbpool has been written by Mike Sweet from the CUPS folks. It is included and ships with Samba. It may also be used with print subsystems other than CUPS, to spool jobs to Windows printer shares. To set up printer *winprinter* on CUPS, you need to have a driver for it. Essentially this means to convert the print data on the CUPS/Samba host to a format that the printer can digest (the Windows host is unable to convert any files you may send). This also means you should be able to print to the printer if it were hooked directly at your Samba/CUPS host. For troubleshooting purposes, this is what you should do to determine if that part of the process chain is in order. Then proceed to fix the network connection/authentication to the Windows host, and so on.

To install a printer with the *smb* backend on CUPS, use this command:

```
root# lpadmin -p winprinter -v smb://WINDOWSNETBIOSNAME/printersharename \  
-P /path/to/PPD
```

The PPD must be able to direct CUPS to generate the print data for the target model. For PostScript printers, just use the PPD that would be used with the Windows NT PostScript driver. But what can you do if the printer is only accessible with a password? Or if the printer's host is part of another workgroup? This is provided for: You can include the required parameters as part of the *smb://* device-URI like this:

- *smb://WORKGROUP/WINDOWSNETBIOSNAME/printersharename*
- *smb://username:password@WORKGROUP/WINDOWSNETBIOSNAME/printersharename*
- *smb://username:password@WINDOWSNETBIOSNAME/printersharename*

Note that the device-URI will be visible in the process list of the Samba server (e.g., when someone uses the **ps -aux** command on Linux), even if the username and passwords are sanitized before they get written into the log files. So this is an inherently insecure option, however, it is the only one. Don't use it if you want to protect your passwords. Better share the printer in a way that does not require a password! Printing will only work if you have a working netbios name resolution up and running. Note that this is a feature of CUPS and you do not necessarily need to have *smbd* running.

18.18 More CUPS-Filtering Chains

The following diagrams reveal how CUPS handles print jobs.

CUPS in and of itself has this (general) filter chain (italic letters are file-formats or MIME types, other are filters (this is true for pre-1.1.15 of pre-4.3 versions of CUPS and ESP PrintPro):

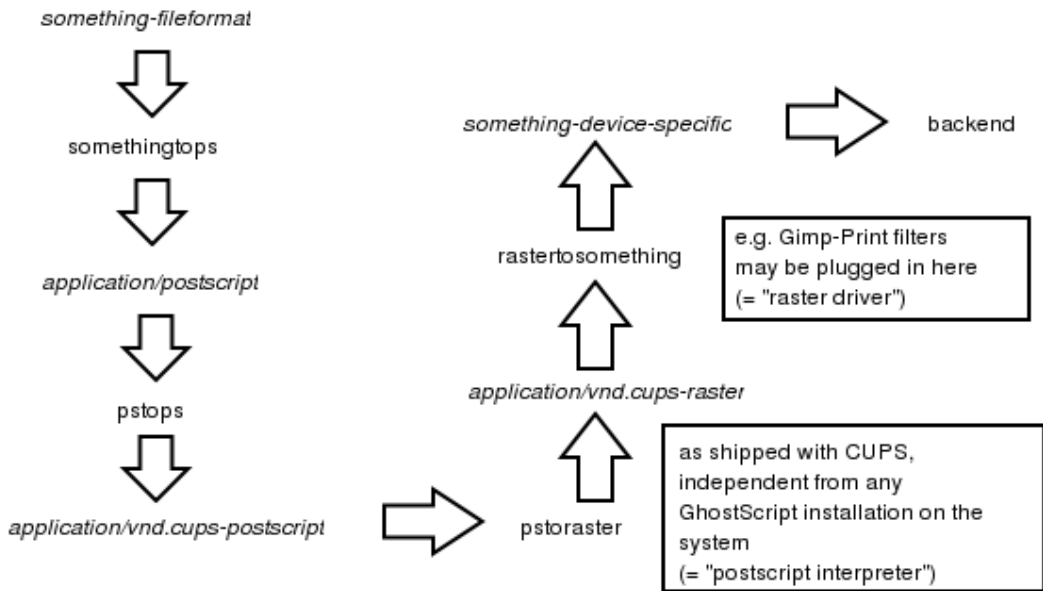


Figure 18.17. Filtering chain 1.

18.19 Common Errors

18.19.1 Windows 9x/ME Client Can't Install Driver

For Windows 9x/ME, clients require the printer names to be eight characters (or “8 plus 3 chars suffix”) max; otherwise, the driver files will not get transferred when you want to download them from Samba.

18.19.2 “*cupsaddsmb*” Keeps Asking for Root Password in Never-ending Loop

Have you *security* = user? Have you used **smbpasswd** to give root a Samba account? You can do two things: open another terminal and execute **smbpasswd -a root** to create the account and continue entering the password into the first terminal. Or break out of the loop by pressing ENTER twice (without trying to type a password).

18.19.3 “*cupsaddsmb*” Errors

The use of “*cupsaddsmb*” gives “No PPD file for printer...” Message While PPD File Is Present. What might the problem be?

Have you enabled printer sharing on CUPS? This means: Do you have a `<Location /printers>...</Location>` section in CUPS server's `cupsd.conf` that does not deny access to the host you run “`cupsaddsmb`” from? It *could* be an issue if you use `cupsaddsmb` remotely, or if you use it with a `-h` parameter: `cupsaddsmb -H sambaserver -h cupsserver -v printername`.

Is your `TempDir` directive in `cupsd.conf` set to a valid value and is it writeable?

18.19.4 Client Can't Connect to Samba Printer

Use `smbstatus` to check which user you are from Samba's point of view. Do you have the privileges to write into the `[print$]` share?

18.19.5 New Account Reconnection from Windows 200x/XP Troubles

Once you are connected as the wrong user (for example, as `nobody`, which often occurs if you have `map to guest = bad user`), Windows Explorer will not accept an attempt to connect again as a different user. There will not be any byte transferred on the wire to Samba, but still you'll see a stupid error message that makes you think Samba has denied access. Use `smbstatus` to check for active connections. Kill the PIDs. You still can't reconnect and you get the dreaded **You can't connect with a second account from the same machine** message, as soon as you are trying. And you do not see any single byte arriving at Samba (see logs; use “`ethereal`”) indicating a renewed connection attempt. Shut all Explorer Windows. This makes Windows forget what it has cached in its memory as established connections. Then reconnect as the right user. The best method is to use a DOS terminal window and *first* do `net use z: \\GANDALF\print$ /user:root`. Check with `smbstatus` that you are connected under a different account. Now open the **Printers** folder (on the Samba server in the **Network Neighborhood**), right-click on the printer in question and select **Connect...**

18.19.6 Avoid Being Connected to the Samba Server as the Wrong User

You see per `smbstatus` that you are connected as user `nobody`; while you want to be `root` or `printeradmin`. This is probably due to `map to guest = bad user`, which silently connects you under the `guest` account when you gave (maybe by accident) an incorrect username. Remove `map to guest`, if you want to prevent this.

18.19.7 Upgrading to CUPS Drivers from Adobe Drivers

This information came from a mailinglist posting regarding problems experienced when upgrading from Adobe drivers to CUPS drivers on Microsoft Windows NT/200x/XP Clients.

First delete all old Adobe-using printers. Then delete all old Adobe drivers. (On Windows 200x/XP, right-click in the background of **Printers** folder, select **Server Properties...**, select tab **Drivers** and delete here).

18.19.8 Can't Use “*cupsaddsmb*” on Samba Server Which Is a PDC

Do you use the “*naked*” root user name? Try to do it this way: `cupsaddsmb -U DO-MAINNAME\\root -v printername >` (note the two backslashes: the first one is required to “*escape*” the second one).

18.19.9 Deleted Windows 200x Printer Driver Is Still Shown

Deleting a printer on the client will not delete the driver too (to verify, right-click on the white background of the **Printers** folder, select **Server Properties** and click on the **Drivers** tab). These same old drivers will be re-used when you try to install a printer with the same name. If you want to update to a new driver, delete the old ones first. Deletion is only possible if no other printer uses the same driver.

18.19.10 Windows 200x/XP “Local Security Policies”

Local Security Policies may not allow the installation of unsigned drivers. “*Local Security Policies*” may not allow the installation of printer drivers at all.

18.19.11 Administrator Cannot Install Printers for All Local Users

Windows XP handles SMB printers on a “*per-user*” basis. This means every user needs to install the printer himself. To have a printer available for everybody, you might want to use the built-in IPP client capabilities of WinXP. Add a printer with the print path of `http://cupsserver:631/printers/printername`. We're still looking into this one. Maybe a logon script could automatically install printers for all users.

18.19.12 Print Change Notify Functions on NT-clients

For print change, notify functions on NT++ clients. These need to run the **Server** service first (renamed to **File & Print Sharing for MS Networks** in XP).

18.19.13 WinXP-SP1

WinXP-SP1 introduced a Point and Print Restriction Policy (this restriction does not apply to “*Administrator*” or “*Power User*” groups of users). In Group Policy Object Editor, go to **User Configuration** -> **Administrative Templates** -> **Control Panel** -> **Printers**. The policy is automatically set to **Enabled** and the **Users can only Point and Print to machines in their Forest**. You probably need to change it to **Disabled** or **Users can only Point and Print to these servers** to make driver downloads from Samba possible.

18.19.14 Print Options for All Users Can't Be Set on Windows 200x/XP

How are you doing it? I bet the wrong way (it is not easy to find out, though). There are three different ways to bring you to a dialog that *seems* to set everything. All three dialogs

look the same, yet only one of them does what you intend. You need to be Administrator or Print Administrator to do this for all users. Here is how I do in on XP:

A The first wrong way:

- (a) Open the **Printers** folder.
- (b) Right-click on the printer (**remotepri~~n~~ter on cupshost**) and select in context menu **Printing Preferences...**
- (c) Look at this dialog closely and remember what it looks like.

B The second wrong way:

- (a) Open the **Printers** folder.
- (b) Right-click on the printer (**remotepri~~n~~ter on cupshost**) and select the context menu **Properties**.
- (c) Click on the **General** tab.
- (d) Click on the button **Printing Preferences...**
- (e) A new dialog opens. Keep this dialog open and go back to the parent dialog.

C The third, and the correct way:

- (a) Open the **Printers** folder.
- (b) Click on the **Advanced** tab. (If everything is “*grayed out*,” then you are not logged in as a user with enough privileges).
- (c) Click on the **Printing Defaults...** button.
- (d) On any of the two new tabs, click on the **Advanced...** button.
- (e) A new dialog opens. Compare this one to the other identical looking one from “*B.5*” or *A.3*”.

Do you see any difference? I don’t either. However, only the last one, which you arrived at with steps “*C.1.-6*,” will save any settings permanently and be the defaults for new users. If you want all clients to get the same defaults, you need to conduct these steps *as Administrator* (*printer admin* in *smb.conf*) *before* a client downloads the driver (the clients can later set their own *per-user defaults* by following the procedures *A* or *B* above).

18.19.15 Most Common Blunders in Driver Settings on Windows Clients

Don’t use *Optimize for Speed*, but use *Optimize for Portability* instead (Adobe PS Driver). Don’t use *Page Independence: No*: always settle with *Page Independence: Yes* (Microsoft PS Driver and CUPS PS Driver for Windows NT/200x/XP). If there are problems with fonts, use *Download as Softfont into printer* (Adobe PS Driver). For **TrueType Download Options** choose *Outline*. Use PostScript Level 2, if you are having trouble with a non-PS printer and if there is a choice.

18.19.16 cupsaddsmb Does Not Work with Newly Installed Printer

Symptom: The last command of **cupsaddsmb** does not complete successfully: **cmd = setdriver printername printername** result was `NT_STATUS_UNSUCCESSFUL` then possibly the printer was not yet recognized by Samba. Did it show up in Network Neighborhood? Did it show up in `rpcclient hostname -c 'enumprinters'`? Restart `smbd` (or send a **kill -HUP** to all processes listed by `smbstatus` and try again.

18.19.17 Permissions on /var/spool/samba/ Get Reset After Each Reboot

Have you ever by accident set the CUPS spool directory to the same location? (*RequestRoot /var/spool/samba/* in `cupsd.conf` or the other way round: `/var/spool/cups/` is set as *path* in the `[printers]` section). These *must* be different. Set

RequestRoot /var/spool/cups/ in `cupsd.conf` and *path = /var/spool/samba* in the `[printers]` section of `smb.conf`. Otherwise `cupsd` will sanitize permissions to its spool directory with each restart and printing will not work reliably.

18.19.18 Print Queue Called “lp” Mis-handles Print Jobs

In this case a print queue called “*lp*” intermittently swallows jobs and spits out completely different ones from what was sent.

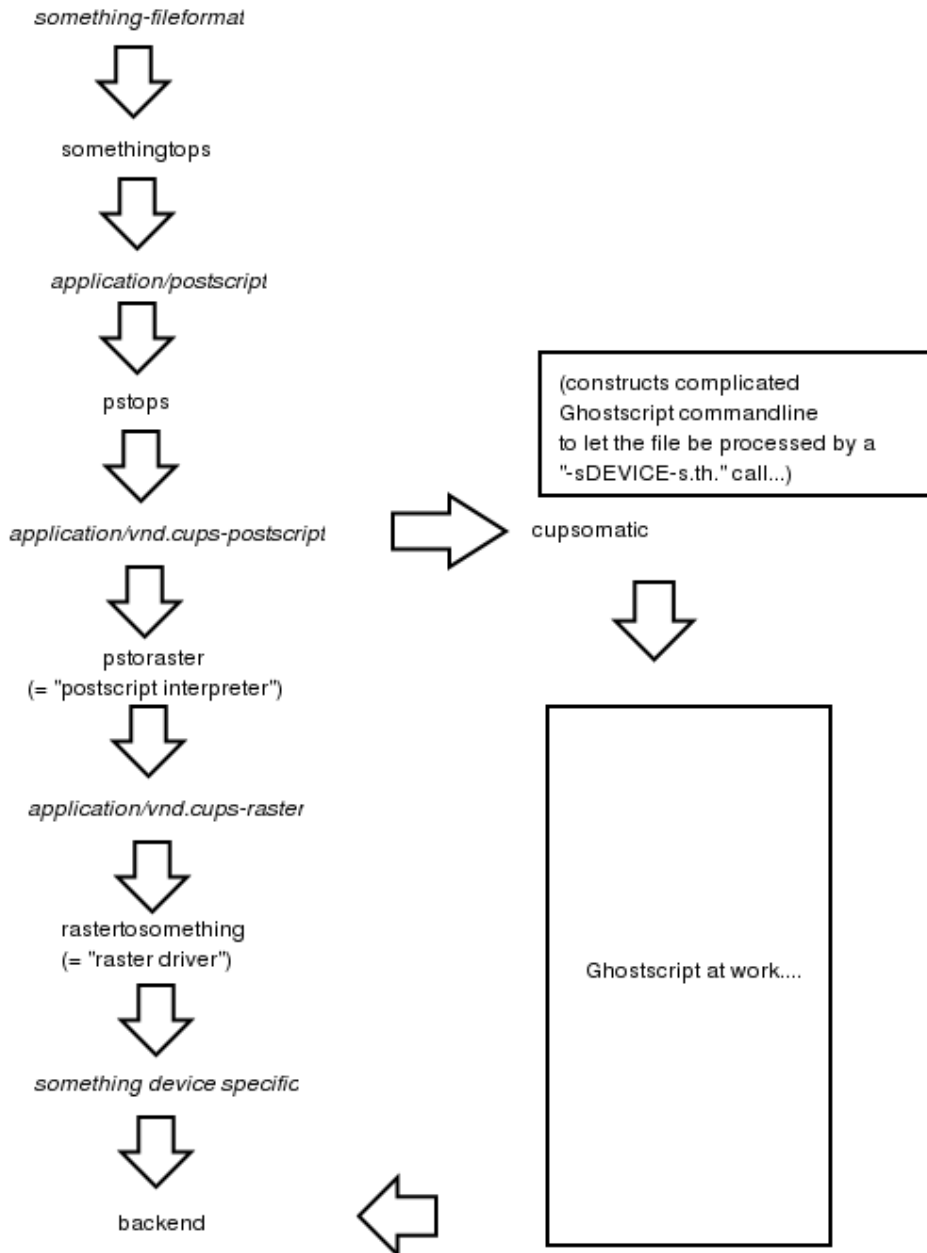
It is a bad idea to name any printer “*lp*”. This is the traditional UNIX name for the default printer. CUPS may be set up to do an automatic creation of Implicit Classes. This means, to group all printers with the same name to a pool of devices, and load-balancing the jobs across them in a round-robin fashion. Chances are high that someone else has a printer named “*lp*” too. You may receive his jobs and send your own to his device unwittingly. To have tight control over the printer names, set *BrowseShortNames No*. It will present any printer as *printername@cupshost* and then gives you better control over what may happen in a large networked environment.

18.19.19 Location of Adobe PostScript Driver Files for “cupsaddsmb”

Use `smbclient` to connect to any Windows box with a shared PostScript printer: `smbclient //windowsbox/print/$ -U guest`. You can navigate to the `W32X86/2` subdir to `mget ADOBE*` and other files or to `WIN40/0` to do the same. Another option is to download the `*.exe` packaged files from the Adobe Web site.

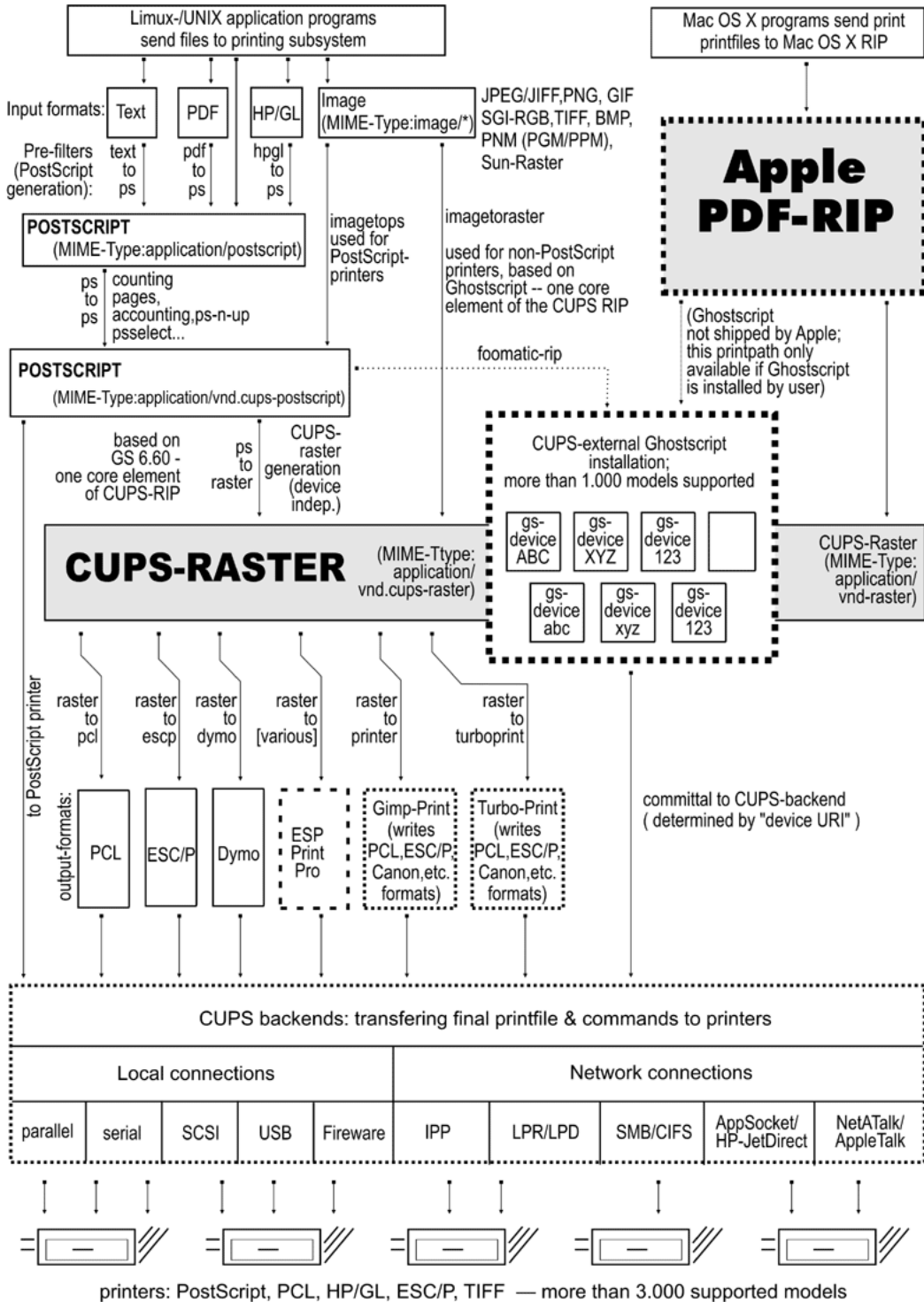
18.20 Overview of the CUPS Printing Processes

A complete overview of the CUPS printing processes can be found in Figure 18.19.



Note, that cupsomatic "kidnaps" the printfile after the application/vnd.cups-postscript stage and deviates it through the CUPS-external, systemwide Ghostscript installation, bypassing the "pstoraster" filter (therefore also bypassing the CUPS-raster-drivers "rastertosomething", and hands the rasterized file directly to the CUPS backend...

Figure 18.18. Filtering chain with cupsomatic



(c) Kurt Pfeifle, Danko Deutschland GmbH graphic design Ciprian Vizitzi

Figure 18.19. CUPS printing overview.

STACKABLE VFS MODULES

19.1 Features and Benefits

Since Samba-3, there is support for stackable VFS (Virtual File System) modules. Samba passes each request to access the UNIX file system through the loaded VFS modules. This chapter covers all the modules that come with the Samba source and references to some external modules.

19.2 Discussion

If not supplied with your platform distribution binary Samba package you may have problems compiling these modules, as shared libraries are compiled and linked in different ways on different systems. They currently have been tested against GNU/Linux and IRIX.

To use the VFS modules, create a share similar to the one below. The important parameter is the *vfs objects* parameter where you can list one or more VFS modules by name. For example, to log all access to files and put deleted files in a recycle bin, see Example 19.1.

Example 19.1. smb.conf with VFS modules

```
[audit]
comment = Audited /data directory
path = /data
vfs objects = audit recycle
writeable = yes
browseable = yes
```

The modules are used in the order in which they are specified.

Samba will attempt to load modules from the `/lib` directory in the root directory of the Samba installation (usually `/usr/lib/samba/vfs` or `/usr/local/samba/lib/vfs`).

Some modules can be used twice for the same share. This can be done using a configuration similar to the one shown in Example 19.2.

Example 19.2. smb.conf with multiple VFS modules

```
[test]
comment = VFS TEST
path = /data
writeable = yes
browseable = yes
vfs objects = example:example1 example example:test
example1: parameter = 1
example: parameter = 5
test: parameter = 7
```

19.3 Included Modules

19.3.1 audit

A simple module to audit file access to the syslog facility. The following operations are logged:

- share
- connect/disconnect
- directory opens/create/remove
- file open/close/rename/unlink/chmod

19.3.2 extd_audit

This module is identical with the **audit** module above except that it sends audit logs to both syslog as well as the **smbd** log files. The *log level* for this module is set in the **smb.conf** file.

Valid settings and the information that will be recorded are shown in Table 19.1.

Table 19.1. Extended Auditing Log Information

Log Level	Log Details - File and Directory Operations
0	Creation / Deletion
1	Create / Delete / Rename / Permission Changes
2	Create / Delete / Rename / Perm Change / Open / Close

19.3.3 fake_perms

This module was created to allow Roaming Profile files and directories to be set (on the Samba server under UNIX) as read only. This module will, if installed on the Profiles share, report to the client that the Profile files and directories are writable. This satisfies the client even though the files will never be overwritten as the client logs out or shuts down.

19.3.4 recycle

A Recycle Bin-like module. Where used, unlink calls will be intercepted and files moved to the recycle directory instead of being deleted. This gives the same effect as the **Recycle Bin** on Windows computers.

The **Recycle Bin** will not appear in Windows Explorer views of the network file system (share) nor on any mapped drive. Instead, a directory called `.recycle` will be automatically created when the first file is deleted. Users can recover files from the `.recycle` directory. If the `recycle:keeptree` has been specified, deleted files will be found in a path identical with that from which the file was deleted.

Supported options for the **recycle** module are as follow:

recycle:repository — Relative path of the directory where deleted files should be moved.

recycle:keeptree — Specifies whether the directory structure should be kept or if the files in the directory that is being deleted should be kept separately in the recycle bin.

recycle:versions — If this option is set, two files with the same name that are deleted will both be kept in the recycle bin. Newer deleted versions of a file will be called “*Copy #x of filename*”.

recycle:touch — Specifies whether a file’s access date should be touched when the file is moved to the recycle bin.

recycle:maxsize — Files that are larger than the number of bytes specified by this parameter will not be put into the recycle bin.

recycle:exclude — List of files that should not be put into the recycle bin when deleted, but deleted in the regular way.

recycle:exclude_dir — Contains a list of directories. When files from these directories are deleted, they are not put into the recycle bin but are deleted in the regular way.

recycle:noversions — Opposite of `recycle:versions`. If both options are specified, this one takes precedence.

19.3.5 netatalk

A netatalk module will ease co-existence of Samba and netatalk file sharing services.

Advantages compared to the old netatalk module:

- Does not care about creating `.AppleDouble` forks, just keeps them in sync.

- If a share in `smb.conf` does not contain `.AppleDouble` item in `hide` or `veto` list, it will be added automatically.

19.4 VFS Modules Available Elsewhere

This section contains a listing of various other VFS modules that have been posted but do not currently reside in the Samba CVS tree for one reason or another (e.g., it is easy for the maintainer to have his or her own CVS tree).

No statements about the stability or functionality of any module should be implied due to its presence here.

19.4.1 DatabaseFS

URL: <http://www.css.tayloru.edu/~elorimer/databasefs/index.php>

By Eric Lorimer.¹

I have created a VFS module that implements a fairly complete read-only filesystem. It presents information from a database as a filesystem in a modular and generic way to allow different databases to be used (originally designed for organizing MP3s under directories such as “*Artists*,” “*Song Keywords*,” and so on. I have since easily applied it to a student roster database.) The directory structure is stored in the database itself and the module makes no assumptions about the database structure beyond the table it requires to run.

Any feedback would be appreciated: comments, suggestions, patches, and so on. If nothing else, hopefully it might prove useful for someone else who wishes to create a virtual filesystem.

19.4.2 vscan

URL: <http://www.openantivirus.org/>

`samba-vscan` is a proof-of-concept module for Samba, which uses the VFS (virtual file system) features of Samba 2.2.x/3.0 alphaX. Of course, Samba has to be compiled with VFS support. `samba-vscan` supports various virus scanners and is maintained by Rainer Link.

¹<mailto:elorimer@css.tayloru.edu>

WINBIND: USE OF DOMAIN ACCOUNTS

20.1 Features and Benefits

Integration of UNIX and Microsoft Windows NT through a unified logon has been considered a “*holy grail*” in heterogeneous computing environments for a long time.

There is one other facility without which UNIX and Microsoft Windows network interoperability would suffer greatly. It is imperative that there be a mechanism for sharing files across UNIX systems and to be able to assign domain user and group ownerships with integrity.

winbind is a component of the Samba suite of programs that solves the unified logon problem. Winbind uses a UNIX implementation of Microsoft RPC calls, Pluggable Authentication Modules, and the Name Service Switch to allow Windows NT domain users to appear and operate as UNIX users on a UNIX machine. This chapter describes the Winbind system, explaining the functionality it provides, how it is configured, and how it works internally.

Winbind provides three separate functions:

- Authentication of user credentials (via PAM).
- Identity resolution (via NSS).
- Winbind maintains a database called `winbind_idmap.tdb` in which it stores mappings between UNIX UIDs / GIDs and NT SIDs. This mapping is used only for users and groups that do not have a local UID/GID. It stores the UID/GID allocated from the `idmap uid/gid` range that it has mapped to the NT SID. If *idmap backend* has been specified as `ldapsam:url` then instead of using a local mapping Winbind will obtain this information from the LDAP database.

NOTE



If **winbindd** is not running, **smbd** (which calls **winbindd**) will fall back to using purely local information from `/etc/passwd` and `/etc/group` and no dynamic mapping will be used.

20.2 Introduction

It is well known that UNIX and Microsoft Windows NT have different models for representing user and group information and use different technologies for implementing them. This fact has made it difficult to integrate the two systems in a satisfactory manner.

One common solution in use today has been to create identically named user accounts on both the UNIX and Windows systems and use the Samba suite of programs to provide file and print services between the two. This solution is far from perfect, however, as adding and deleting users on both sets of machines becomes a chore and two sets of passwords are required — both of which can lead to synchronization problems between the UNIX and Windows systems and confusion for users.

We divide the unified logon problem for UNIX machines into three smaller problems:

- Obtaining Windows NT user and group information.
- Authenticating Windows NT users.
- Password changing for Windows NT users.

Ideally, a prospective solution to the unified logon problem would satisfy all the above components without duplication of information on the UNIX machines and without creating additional tasks for the system administrator when maintaining users and groups on either system. The Winbind system provides a simple and elegant solution to all three components of the unified logon problem.

20.3 What Winbind Provides

Winbind unifies UNIX and Windows NT account management by allowing a UNIX box to become a full member of an NT domain. Once this is done the UNIX box will see NT users and groups as if they were “*native*” UNIX users and groups, allowing the NT domain to be used in much the same manner that NIS+ is used within UNIX-only environments.

The end result is that whenever any program on the UNIX machine asks the operating system to lookup a user or group name, the query will be resolved by asking the NT Domain Controller for the specified domain to do the lookup. Because Winbind hooks into the operating system at a low level (via the NSS name resolution modules in the C library), this redirection to the NT Domain Controller is completely transparent.

Users on the UNIX machine can then use NT user and group names as they would “*native*” UNIX names. They can chown files so they are owned by NT domain users or even login to the UNIX machine and run a UNIX X-Window session as a domain user.

The only obvious indication that Winbind is being used is that user and group names take the form `DOMAIN\user` and `DOMAIN\group`. This is necessary as it allows Winbind to determine that redirection to a Domain Controller is wanted for a particular lookup and which trusted domain is being referenced.

Additionally, Winbind provides an authentication service that hooks into the Pluggable Authentication Modules (PAM) system to provide authentication via an NT domain to any PAM-enabled applications. This capability solves the problem of synchronizing passwords between systems since all passwords are stored in a single location (on the Domain Controller).

20.3.1 Target Uses

Winbind is targeted at organizations that have an existing NT-based domain infrastructure into which they wish to put UNIX workstations or servers. Winbind will allow these organizations to deploy UNIX workstations without having to maintain a separate account infrastructure. This greatly simplifies the administrative overhead of deploying UNIX workstations into an NT-based organization.

Another interesting way in which we expect Winbind to be used is as a central part of UNIX-based appliances. Appliances that provide file and print services to Microsoft-based networks will be able to use Winbind to provide seamless integration of the appliance into the domain.

20.4 How Winbind Works

The Winbind system is designed around a client/server architecture. A long running **winbindd** daemon listens on a UNIX domain socket waiting for requests to arrive. These requests are generated by the NSS and PAM clients and is processed sequentially.

The technologies used to implement Winbind are described in detail below.

20.4.1 Microsoft Remote Procedure Calls

Over the last few years, efforts have been underway by various Samba Team members to decode various aspects of the Microsoft Remote Procedure Call (MSRPC) system. This system is used for most network-related operations between Windows NT machines including remote management, user authentication and print spooling. Although initially this work was done to aid the implementation of Primary Domain Controller (PDC) functionality in Samba, it has also yielded a body of code that can be used for other purposes.

Winbind uses various MSRPC calls to enumerate domain users and groups and to obtain detailed information about individual users or groups. Other MSRPC calls can be used to authenticate NT domain users and to change user passwords. By directly querying a

Windows PDC for user and group information, Winbind maps the NT account information onto UNIX user and group names.

20.4.2 Microsoft Active Directory Services

Since late 2001, Samba has gained the ability to interact with Microsoft Windows 2000 using its “*Native Mode*” protocols, rather than the NT4 RPC services. Using LDAP and Kerberos, a Domain Member running Winbind can enumerate users and groups in exactly the same way as a Windows 200x client would, and in so doing provide a much more efficient and effective Winbind implementation.

20.4.3 Name Service Switch

The Name Service Switch, or NSS, is a feature that is present in many UNIX operating systems. It allows system information such as hostnames, mail aliases and user information to be resolved from different sources. For example, a standalone UNIX workstation may resolve system information from a series of flat files stored on the local filesystem. A networked workstation may first attempt to resolve system information from local files, and then consult an NIS database for user information or a DNS server for hostname information.

The NSS application programming interface allows Winbind to present itself as a source of system information when resolving UNIX usernames and groups. Winbind uses this interface, and information obtained from a Windows NT server using MSRPC calls to provide a new source of account enumeration. Using standard UNIX library calls, one can enumerate the users and groups on a UNIX machine running Winbind and see all users and groups in a NT domain plus any trusted domain as though they were local users and groups.

The primary control file for NSS is `/etc/nsswitch.conf`. When a UNIX application makes a request to do a lookup, the C library looks in `/etc/nsswitch.conf` for a line that matches the service type being requested, for example the “*passwd*” service type is used when user or group names are looked up. This config line specifies which implementations of that service should be tried and in what order. If the `passwd` config line is:

```
passwd: files example
```

then the C library will first load a module called `/lib/libnss_files.so` followed by the module `/lib/libnss_example.so`. The C library will dynamically load each of these modules in turn and call resolver functions within the modules to try to resolve the request. Once the request is resolved, the C library returns the result to the application.

This NSS interface provides an easy way for Winbind to hook into the operating system. All that needs to be done is to put `libnss_winbind.so` in `/lib/` then add “*winbind*” into `/etc/nsswitch.conf` at the appropriate place. The C library will then call Winbind to resolve user and group names.

20.4.4 Pluggable Authentication Modules

Pluggable Authentication Modules, also known as PAM, is a system for abstracting authentication and authorization technologies. With a PAM module it is possible to specify different authentication methods for different system applications without having to recompile these applications. PAM is also useful for implementing a particular policy for authorization. For example, a system administrator may only allow console logins from users stored in the local password file but only allow users resolved from a NIS database to log in over the network.

Winbind uses the authentication management and password management PAM interface to integrate Windows NT users into a UNIX system. This allows Windows NT users to log in to a UNIX machine and be authenticated against a suitable Primary Domain Controller. These users can also change their passwords and have this change take effect directly on the Primary Domain Controller.

PAM is configured by providing control files in the directory `/etc/pam.d/` for each of the services that require authentication. When an authentication request is made by an application, the PAM code in the C library looks up this control file to determine what modules to load to do the authentication check and in what order. This interface makes adding a new authentication service for Winbind very easy. All that needs to be done is that the `pam.winbind.so` module is copied to `/lib/security/` and the PAM control files for relevant services are updated to allow authentication via Winbind. See the PAM documentation in Chapter 24, *PAM-Based Distributed Authentication* for more information.

20.4.5 User and Group ID Allocation

When a user or group is created under Windows NT/200x it is allocated a numerical relative identifier (RID). This is slightly different from UNIX which has a range of numbers that are used to identify users, and the same range in which to identify groups. It is Winbind's job to convert RIDs to UNIX ID numbers and vice versa. When Winbind is configured, it is given part of the UNIX user ID space and a part of the UNIX group ID space in which to store Windows NT users and groups. If a Windows NT user is resolved for the first time, it is allocated the next UNIX ID from the range. The same process applies for Windows NT groups. Over time, Winbind will have mapped all Windows NT users and groups to UNIX user IDs and group IDs.

The results of this mapping are stored persistently in an ID mapping database held in a tdb database). This ensures that RIDs are mapped to UNIX IDs in a consistent way.

20.4.6 Result Caching

An active system can generate a lot of user and group name lookups. To reduce the network cost of these lookups, Winbind uses a caching scheme based on the SAM sequence number supplied by NT Domain Controllers. User or group information returned by a PDC is cached by Winbind along with a sequence number also returned by the PDC. This sequence number is incremented by Windows NT whenever any user or group information is modified. If a cached entry has expired, the sequence number is requested from the PDC and compared against the sequence number of the cached entry. If the sequence numbers do not match,

then the cached information is discarded and up-to-date information is requested directly from the PDC.

20.5 Installation and Configuration

20.5.1 Introduction

This section describes the procedures used to get Winbind up and running. Winbind is capable of providing access and authentication control for Windows Domain users through an NT or Windows 200x PDC for regular services, such as telnet and ftp, as well for Samba services.

- *Why should I do this?*

This allows the Samba administrator to rely on the authentication mechanisms on the Windows NT/200x PDC for the authentication of Domain Members. Windows NT/200x users no longer need to have separate accounts on the Samba server.

- *Who should be reading this document?*

This document is designed for system administrators. If you are implementing Samba on a file server and wish to (fairly easily) integrate existing Windows NT/200x users from your PDC onto the Samba server, this document is for you.

20.5.2 Requirements

If you have a Samba configuration file that you are currently using, *BACK IT UP!* If your system already uses PAM, *back up the /etc/pam.d directory contents!* If you haven't already made a boot disk, *MAKE ONE NOW!*

Messing with the PAM configuration files can make it nearly impossible to log in to your machine. That's why you want to be able to boot back into your machine in single user mode and restore your `/etc/pam.d` back to the original state they were in if you get frustrated with the way things are going.

The latest version of Samba-3 includes a functioning winbindd daemon. Please refer to the main Samba Web page¹ or, better yet, your closest Samba mirror site for instructions on downloading the source code.

To allow domain users the ability to access Samba shares and files, as well as potentially other services provided by your Samba machine, PAM must be set up properly on your machine. In order to compile the Winbind modules, you should have at least the PAM development libraries installed on your system. Please refer the PAM web site <http://www.kernel.org/pub/linux/libs/pam/>².

¹<http://samba.org/>

²<http://www.kernel.org/pub/linux/libs/pam/>

20.5.3 Testing Things Out

Before starting, it is probably best to kill off all the Samba-related daemons running on your server. Kill off all `smbd`, `nmbd`, and `winbindd` processes that may be running. To use PAM, make sure that you have the standard PAM package that supplies the `/etc/pam.d` directory structure, including the PAM modules that are used by PAM-aware services, several pam libraries, and the `/usr/doc` and `/usr/man` entries for pam. Winbind built better in Samba if the `pam-devel` package is also installed. This package includes the header files needed to compile PAM-aware applications.

20.5.3.1 Configure `nsswitch.conf` and the Winbind Libraries on Linux and Solaris

PAM is a standard component of most current generation UNIX/Linux systems. Unfortunately, few systems install the `pam-devel` libraries that are needed to build PAM-enabled Samba. Additionally, Samba-3 may auto-install the Winbind files into their correct locations on your system, so before you get too far down the track be sure to check if the following configuration is really necessary. You may only need to configure `/etc/nsswitch.conf`.

The libraries needed to run the `winbindd` daemon through `nsswitch` need to be copied to their proper locations:

```
root# cp ../samba/source/nsswitch/libnss_winbind.so /lib
```

I also found it necessary to make the following symbolic link:

```
root# ln -s /lib/libnss_winbind.so /lib/libnss_winbind.so.2
```

And, in the case of Sun Solaris:

```
root# ln -s /usr/lib/libnss_winbind.so /usr/lib/libnss_winbind.so.1
root# ln -s /usr/lib/libnss_winbind.so /usr/lib/nss_winbind.so.1
root# ln -s /usr/lib/libnss_winbind.so /usr/lib/nss_winbind.so.2
```

Now, as root you need to edit `/etc/nsswitch.conf` to allow user and group entries to be visible from the `winbindd` daemon. My `/etc/nsswitch.conf` file look like this after editing:

```
passwd:      files winbind
shadow:      files
group:       files winbind
```

The libraries needed by the `winbindd` daemon will be automatically entered into the `ldconfig` cache the next time your system reboots, but it is faster (and you do not need to reboot) if you do it manually:

```
root#/sbin/ldconfig -v | grep winbind
```

This makes `libnss_winbind` available to `winbindd` and echos back a check to you.

20.5.3.2 NSS Winbind on AIX

(This section is only for those running AIX.)

The Winbind AIX identification module gets built as `libnss_winbind.so` in the `nsswitch` directory of the Samba source. This file can be copied to `/usr/lib/security`, and the AIX naming convention would indicate that it should be named `WINBIND`. A stanza like the following:

```
WINBIND:
    program = /usr/lib/security/WINBIND
    options = authonly
```

can then be added to `/usr/lib/security/methods.cfg`. This module only supports identification, but there have been success reports using the standard Winbind PAM module for authentication. Use caution configuring loadable authentication modules since you can make it impossible to logon to the system. More information about the AIX authentication module API can be found at “*Kernel Extensions and Device Support Programming Concepts for AIX*” in Chapter 18 (John, there is no section like this in 18). Loadable Authentication Module Programming Interface³ and more information on administering the modules can be found at “*System Management Guide: Operating System and Devices*.”⁴

20.5.3.3 Configure `smb.conf`

Several parameters are needed in the `smb.conf` file to control the behavior of `winbindd`. These are described in more detail in the `winbindd(8)` man page. My `smb.conf` file, as shown in Example 20.1, was modified to include the necessary entries in the `[global]` section.

20.5.3.4 Join the Samba Server to the PDC Domain

Enter the following command to make the Samba server join the PDC domain, where *DOMAIN* is the name of your Windows domain and *Administrator* is a domain user who has administrative privileges in the domain.

```
root#/usr/local/samba/bin/net rpc join -S PDC -U Administrator
```

The proper response to the command should be: “*Joined the domain DOMAIN*” where *DOMAIN* is your DOMAIN name.

20.5.3.5 Starting and Testing the `winbindd` Daemon

Eventually, you will want to modify your Samba startup script to automatically invoke the `winbindd` daemon when the other parts of Samba start, but it is possible to test out just

³http://publibn.boulder.ibm.com/doc.link/en_US/a_doc.lib/aixprgpd/kernextc/sec.load_mod.htm

⁴http://publibn.boulder.ibm.com/doc.link/en_US/a_doc.lib/aixbman/baseadm/iandaadmin.htm

Example 20.1. smb.conf for Winbind set-up

```
[global]
# separate domain and username with '+', like DOMAIN+username
winbind separator = +
# use uids from 10000 to 20000 for domain users
idmap uid = 10000-20000
# use gids from 10000 to 20000 for domain groups
idmap gid = 10000-20000
# allow enumeration of winbind users and groups
winbind enum users = yes
winbind enum groups = yes
# give winbind users a real shell (only needed if they have telnet access)
template homedir = /home/winnt/%D/%U
template shell = /bin/bash
```

the Winbind portion first. To start up Winbind services, enter the following command as root:

```
root#/usr/local/samba/bin/winbindd
```

NOTE

The above assumes that Samba has been installed in the `/usr/local/samba` directory tree. You may need to search for the location of Samba files if this is not the location of **winbindd** on your system.

Winbindd can now also run in “*dual daemon mode*”. This will make it run as two processes. The first will answer all requests from the cache, thus making responses to clients faster. The other will update the cache for the query that the first has just responded. The advantage of this is that responses stay accurate and are faster. You can enable dual daemon mode by adding `-B` to the commandline:

```
root#/usr/local/samba/bin/winbindd -B
```

I’m always paranoid and like to make sure the daemon is really running.

```
root#ps -ae | grep winbindd
```

This command should produce output like this, if the daemon is running you would expect to see a report something like this:

```
3025 ?          00:00:00 winbindd
```

Now, for the real test, try to get some information about the users on your PDC:

```
root#/usr/local/samba/bin/wbinfo -u
```

This should echo back a list of users on your Windows users on your PDC. For example, I get the following response:

```
CEO+Administrator
CEO+burdell
CEO+Guest
CEO+jt-ad
CEO+krbtgt
CEO+TsInternetUser
```

Obviously, I have named my domain “*CEO*” and my *winbind separator* is “+”.

You can do the same sort of thing to get group information from the PDC:

```
root# /usr/local/samba/bin/wbinfo -g
CEO+Domain Admins
CEO+Domain Users
CEO+Domain Guests
CEO+Domain Computers
CEO+Domain Controllers
CEO+Cert Publishers
CEO+Schema Admins
CEO+Enterprise Admins
CEO+Group Policy Creator Owners
```

The function **getent** can now be used to get unified lists of both local and PDC users and groups. Try the following command:

```
root#getent passwd
```

You should get a list that looks like your `/etc/passwd` list followed by the domain users with their new UIDs, GIDs, home directories and default shells.

The same thing can be done for groups with the command:

```
root#getent group
```

20.5.3.6 Fix the `init.d` Startup Scripts

Linux The `winbindd` daemon needs to start up after the `smbd` and `nmbd` daemons are running. To accomplish this task, you need to modify the startup scripts of your system. They are located at `/etc/init.d/smb` in Red Hat Linux and they are located in `/etc/init.d/samba` in Debian Linux. Edit your script to add commands to invoke this daemon in the proper sequence. My startup script starts up `smbd`, `nmbd`, and `winbindd` from the `/usr/local/samba/bin` directory directly. The **start** function in the script looks like this:

```

start() {
    KIND="SMB"
    echo -n "$Starting $KIND services: "
    daemon /usr/local/samba/bin/smbd $SMBDOPTIONS
    RETVAL=$?
    echo
    KIND="NMB"
    echo -n "$Starting $KIND services: "
    daemon /usr/local/samba/bin/nmbd $NMBDOPTIONS
    RETVAL2=$?
    echo
    KIND="Winbind"
    echo -n "$Starting $KIND services: "
    daemon /usr/local/samba/bin/winbindd
    RETVAL3=$?
    echo
    [ $RETVAL -eq 0 -a $RETVAL2 -eq 0 -a $RETVAL3 -eq 0 ] && \
touch /var/lock/subsys/smb || RETVAL=1
    return $RETVAL
}

```

If you would like to run winbindd in dual daemon mode, replace the line :

```
daemon /usr/local/samba/bin/winbindd
```

in the example above with:

```
daemon /usr/local/samba/bin/winbindd -B
```

The **stop** function has a corresponding entry to shut down the services and looks like this:

```

stop() {
    KIND="SMB"
    echo -n "$Shutting down $KIND services: "
    killproc smbd
    RETVAL=$?
    echo
    KIND="NMB"
    echo -n "$Shutting down $KIND services: "
    killproc nmbd
    RETVAL2=$?
    echo

```

```

    KIND="Winbind"
    echo -n $"Shutting down $KIND services: "
    killproc winbindd
    RETVAL3=$?
    [ $RETVAL -eq 0 -a $RETVAL2 -eq 0 -a $RETVAL3 -eq 0 ] && \
rm -f /var/lock/subsys/smb
    echo ""
    return $RETVAL
}

```

Solaris Winbind does not work on Solaris 9, see Section 36.6.2 for details.

On Solaris, you need to modify the `/etc/init.d/samba.server` startup script. It usually only starts `smbd` and `nmbd` but should now start `winbindd`, too. If you have Samba installed in `/usr/local/samba/bin`, the file could contains something like this:

```

##
## samba.server
##

if [ ! -d /usr/bin ]
then
    # /usr not mounted
    exit
fi

killproc() {
    # kill the named process(es)
    pid='/usr/bin/ps -e |
        /usr/bin/grep -w $1 |
        /usr/bin/sed -e 's/^ *//' -e 's/ .*//''
    [ "$pid" != "" ] && kill $pid
}

# Start/stop processes required for Samba server

case "$1" in

'start')
#
# Edit these lines to suit your installation (paths, workgroup, host)
#
echo Starting SMBD
    /usr/local/samba/bin/smbd -D -s \
    /usr/local/samba/smb.conf

echo Starting NMBD
    /usr/local/samba/bin/nmbd -D -l \
    /usr/local/samba/var/log -s /usr/local/samba/smb.conf

```

```

echo Starting Winbind Daemon
  /usr/local/samba/bin/winbindd
  ;;

'stop')
  killproc nmbd
  killproc smb
  killproc winbindd
  ;;

*)
  echo "Usage: /etc/init.d/samba.server { start | stop }"
  ;;
esac

```

Again, if you would like to run Samba in dual daemon mode, replace:

```
/usr/local/samba/bin/winbindd
```

in the script above with:

```
/usr/local/samba/bin/winbindd -B
```

Restarting If you restart the `smbd`, `nmbd`, and `winbindd` daemons at this point, you should be able to connect to the Samba server as a Domain Member just as if you were a local user.

20.5.3.7 Configure Winbind and PAM

If you have made it this far, you know that **winbindd** and Samba are working together. If you want to use Winbind to provide authentication for other services, keep reading. The PAM configuration files need to be altered in this step. (Did you remember to make backups of your original `/etc/pam.d` files? If not, do it now.)

You will need a PAM module to use `winbindd` with these other services. This module will be compiled in the `../source/nsswitch` directory by invoking the command:

```
root#make nsswitch/pam_winbind.so
```

from the `../source` directory. The `pam_winbind.so` file should be copied to the location of your other PAM security modules. On my RedHat system, this was the `/lib/security` directory. On Solaris, the PAM security modules reside in `/usr/lib/security`.

```
root#cp ../samba/source/nsswitch/pam_winbind.so /lib/security
```

Linux/FreeBSD-specific PAM configuration The `/etc/pam.d/samba` file does not need to be changed. I just left this file as it was:

```
auth    required    /lib/security/pam_stack.so service=system-auth
account required    /lib/security/pam_stack.so service=system-auth
```

The other services that I modified to allow the use of Winbind as an authentication service were the normal login on the console (or a terminal session), telnet logins, and ftp service. In order to enable these services, you may first need to change the entries in `/etc/xinetd.d` (or `/etc/inetd.conf`). Red Hat Linux 7.1 and later uses the new `xinetd.d` structure, in this case you need to change the lines in `/etc/xinetd.d/telnet` and `/etc/xinetd.d/wu-ftp` from

```
enable = no
```

to:

```
enable = yes
```

For ftp services to work properly, you will also need to either have individual directories for the domain users already present on the server, or change the home directory template to a general directory for all domain users. These can be easily set using the `smb.conf` global entry `template homedir`.

The `/etc/pam.d/ftp` file can be changed to allow Winbind ftp access in a manner similar to the samba file. My `/etc/pam.d/ftp` file was changed to look like this:

```
auth    required    /lib/security/pam_listfile.so item=user sense=deny \
file=/etc/ftpusers onerr=succeed
auth    sufficient  /lib/security/pam_winbind.so
auth    required    /lib/security/pam_stack.so service=system-auth
auth    required    /lib/security/pam_shells.so
account sufficient  /lib/security/pam_winbind.so
account required    /lib/security/pam_stack.so service=system-auth
session required    /lib/security/pam_stack.so service=system-auth
```

The `/etc/pam.d/login` file can be changed nearly the same way. It now looks like this:

```
auth    required    /lib/security/pam_securetty.so
auth    sufficient  /lib/security/pam_winbind.so
auth    sufficient  /lib/security/pam_UNIX.so use_first_pass
auth    required    /lib/security/pam_stack.so service=system-auth
auth    required    /lib/security/pam_nologin.so
account sufficient  /lib/security/pam_winbind.so
```



```

account    required    /lib/security/pam_stack.so service=system-auth
password   required    /lib/security/pam_stack.so service=system-auth
session    required    /lib/security/pam_stack.so service=system-auth
session    optional    /lib/security/pam_console.so

```

In this case, I added the

```
auth sufficient /lib/security/pam_winbind.so
```

lines as before, but also added the

```
required pam_securetty.so
```

above it, to disallow root logins over the network. I also added a

```
sufficient /lib/security/pam_unix.so use_first_pass
```

line after the **winbind.so** line to get rid of annoying double prompts for passwords.

Solaris-specific configuration The `/etc/pam.conf` needs to be changed. I changed this file so my Domain users can logon both locally as well as telnet. The following are the changes that I made. You can customize the `pam.conf` file as per your requirements, but be sure of those changes because in the worst case it will leave your system nearly impossible to boot.

```

#
#ident "@(#)pam.conf 1.14 99/09/16 SMI"
#
# Copyright (c) 1996-1999, Sun Microsystems, Inc.
# All Rights Reserved.
#
# PAM configuration
#
# Authentication management
#
login    auth required    /usr/lib/security/pam_winbind.so
login    auth required    /usr/lib/security/$ISA/pam_UNIX.so.1 try_first_pass
login    auth required    /usr/lib/security/$ISA/pam_dial_auth.so.1 try_first_pass
#
rlogin   auth sufficient  /usr/lib/security/pam_winbind.so
rlogin   auth sufficient  /usr/lib/security/$ISA/pam_rhosts_auth.so.1
rlogin   auth required    /usr/lib/security/$ISA/pam_UNIX.so.1 try_first_pass
#
dtlogin  auth sufficient  /usr/lib/security/pam_winbind.so
dtlogin  auth required    /usr/lib/security/$ISA/pam_UNIX.so.1 try_first_pass
#
rsh      auth required    /usr/lib/security/$ISA/pam_rhosts_auth.so.1
other    auth sufficient  /usr/lib/security/pam_winbind.so
other    auth required    /usr/lib/security/$ISA/pam_UNIX.so.1 try_first_pass
#

```

```

# Account management
#
login account sufficient /usr/lib/security/pam_winbind.so
login account requisite /usr/lib/security/$ISA/pam_roles.so.1
login account required /usr/lib/security/$ISA/pam_UNIX.so.1
#
dtlogin account sufficient /usr/lib/security/pam_winbind.so
dtlogin account requisite /usr/lib/security/$ISA/pam_roles.so.1
dtlogin account required /usr/lib/security/$ISA/pam_UNIX.so.1
#
other account sufficient /usr/lib/security/pam_winbind.so
other account requisite /usr/lib/security/$ISA/pam_roles.so.1
other account required /usr/lib/security/$ISA/pam_UNIX.so.1
#
# Session management
#
other session required /usr/lib/security/$ISA/pam_UNIX.so.1
#
# Password management
#
#other password sufficient /usr/lib/security/pam_winbind.so
other password required /usr/lib/security/$ISA/pam_UNIX.so.1
dtsession auth required /usr/lib/security/$ISA/pam_UNIX.so.1
#
# Support for Kerberos V5 authentication (uncomment to use Kerberos)
#
#rlogin auth optional /usr/lib/security/$ISA/pam_krb5.so.1 try_first_pass
#login auth optional /usr/lib/security/$ISA/pam_krb5.so.1 try_first_pass
#dtlogin auth optional /usr/lib/security/$ISA/pam_krb5.so.1 try_first_pass
#other auth optional /usr/lib/security/$ISA/pam_krb5.so.1 try_first_pass
#dtlogin account optional /usr/lib/security/$ISA/pam_krb5.so.1
#other account optional /usr/lib/security/$ISA/pam_krb5.so.1
#other session optional /usr/lib/security/$ISA/pam_krb5.so.1
#other password optional /usr/lib/security/$ISA/pam_krb5.so.1 try_first_pass

```

I also added a *try_first_pass* line after the *winbind.so* line to get rid of annoying double prompts for passwords.

Now restart your Samba and try connecting through your application that you configured in the *pam.conf*.

20.6 Conclusion

The Winbind system, through the use of the Name Service Switch, Pluggable Authentication Modules, and appropriate Microsoft RPC calls have allowed us to provide seamless integration of Microsoft Windows NT domain users on a UNIX system. The result is a great reduction in the administrative cost of running a mixed UNIX and NT network.


20.7 Common Errors

Winbind has a number of limitations in its current released version that we hope to overcome in future releases:

- Winbind is currently only available for the Linux, Solaris, AIX, and IRIX operating systems, although ports to other operating systems are certainly possible. For such ports to be feasible, we require the C library of the target operating system to support the Name Service Switch and Pluggable Authentication Modules systems. This is becoming more common as NSS and PAM gain support among UNIX vendors.
- The mappings of Windows NT RIDs to UNIX IDs is not made algorithmically and depends on the order in which unmapped users or groups are seen by Winbind. It may be difficult to recover the mappings of RID to UNIX ID mapping if the file containing this information is corrupted or destroyed.
- Currently the Winbind PAM module does not take into account possible workstation and logon time restrictions that may be set for Windows NT users, this is instead up to the PDC to enforce.

20.7.1 NSCD Problem Warning

WARNING



Do not under any circumstances run **nscd** on any system on which **winbindd** is running.

If **nscd** is running on the UNIX/Linux system, then even though NSSWITCH is correctly configured it will not be possible to resolve domain users and groups for file and directory controls.

20.7.2 Winbind Is Not Resolving Users and Groups

*“My smb.conf file is correctly configured. I have specified `idmap uid = 12000`, and `idmap gid = 3000-3500` and **winbind** is running. When I do the following it all works fine.”*

```
root# wbinfo -u
MIDEARTH+maryo
MIDEARTH+jackb
MIDEARTH+ameds
...
MIDEARTH+root
```

```
root# wbinfo -g
MIDEARTH+Domain Users
```

```
MIDEARTH+Domain Admins
MIDEARTH+Domain Guests
...
MIDEARTH+Accounts

root# getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
...
maryo:x:15000:15003:Mary Orville:/home/MIDEARTH/maryo:/bin/false
```

“But the following command just fails:

```
root# chown maryo a_file
chown: ‘maryo’: invalid user
```

This is driving me nuts! What can be wrong?”

Same problem as the one above. Your system is likely running **nscd**, the name service caching daemon. Shut it down, do not restart it! You will find your problem resolved.

ADVANCED NETWORK MANAGEMENT

This section documents peripheral issues that are of great importance to network administrators who want to improve network resource access control, to automate the user environment and to make their lives a little easier.

21.1 Features and Benefits

Often the difference between a working network environment and a well appreciated one can best be measured by the *little things* that make everything work more harmoniously. A key part of every network environment solution is the ability to remotely manage MS Windows workstations, remotely access the Samba server, provide customized logon scripts, as well as other housekeeping activities that help to sustain more reliable network operations.

This chapter presents information on each of these areas. They are placed here, and not in other chapters, for ease of reference.

21.2 Remote Server Administration

“How do I get ‘User Manager’ and ‘Server Manager’?”

Since I do not need to buy an NT4 Server, how do I get the ‘User Manager for Domains’ and the ‘Server Manager’?

Microsoft distributes a version of these tools called `Nexus.exe` for installation on Windows 9x/Me systems. The tools set includes:

- Server Manager
- User Manager for Domains
- Event Viewer

Download the archived file at <ftp://ftp.microsoft.com/Softlib/MSLFILES/NEXUS.EXE>.

The Windows NT 4.0 version of the ‘User Manager for Domains’ and ‘Server Manager’ are available from Microsoft via ftp¹.

21.3 Remote Desktop Management

There are a number of possible remote desktop management solutions that range from free through costly. Do not let that put you off. Sometimes the most costly solution is the most cost effective. In any case, you will need to draw your own conclusions as to which is the best tool in your network environment.

21.3.1 Remote Management from NoMachine.Com

The following information was posted to the Samba mailing list at Apr 3 23:33:50 GMT 2003. It is presented in slightly edited form (with author details omitted for privacy reasons). The entire answer is reproduced below with some comments removed.

“I have a wonderful Linux/Samba server running as pdc for a network. Now I would like to add remote desktop capabilities so users outside could login to the system and get their desktop up from home or another country.”

“Is there a way to accomplish this? Do I need a Windows Terminal Server? Do I need to configure it so it is a member of the domain or a BDC,PDC? Are there any hacks for MS Windows XP to enable remote login even if the computer is in a domain?”

Answer provided: Check out the new offer from NoMachine, “NX” software: <http://www.nomachine.com/>.

It implements an easy-to-use interface to the Remote X protocol as well as incorporating VNC/RFB and rdesktop/RDP into it, but at a speed performance much better than anything you may have ever seen.

Remote X is not new at all, but what they did achieve successfully is a new way of compression and caching technologies that makes the thing fast enough to run even over slow modem/ISDN connections.

I could test drive their (public) Red Hat machine in Italy, over a loaded Internet connection, with enabled thumbnail previews in KDE konqueror which popped up immediately on “mouse-over”. From inside that (remote X) session I started a rdesktop session on another, a Windows XP machine. To test the performance, I played Pinball. I am proud to announce that my score was 631750 points at first try.

NX performs better on my local LAN than any of the other “pure” connection methods I am using from time to time: TightVNC, rdesktop or Remote X. It is even faster than a direct crosslink connection between two nodes.

I even got sound playing from the Remote X app to my local boxes, and had a working “copy’n’paste” from an NX window (running a KDE session in Italy) to my Mozilla mailing agent. These guys are certainly doing something right!

¹<ftp://ftp.microsoft.com/Softlib/MSLFILES/SRVTOOLS.EXE>

I recommend to test drive NX to anybody with a only a passing interest in remote computing <http://www.nomachine.com/testdrive.php>.

Just download the free of charge client software (available for Red Hat, SuSE, Debian and Windows) and be up and running within five minutes (they need to send you your account data, though, because you are assigned a real UNIX account on their testdrive.nomachine.com box.

They plan to get to the point were you can have NX application servers running as a cluster of nodes, and users simply start an NX session locally, and can select applications to run transparently (apps may even run on another NX node, but pretend to be on the same as used for initial login, because it displays in the same window. You also can run it fullscreen, and after a short time you forget that it is a remote session at all).

Now the best thing for last: All the core compression and caching technologies are released under the GPL and available as source code to anybody who wants to build on it! These technologies are working, albeit started from the command line only (and very inconvenient to use in order to get a fully running remote X session up and running.)

To answer your questions:

- You do not need to install a terminal server; XP has RDP support built in.
- NX is much cheaper than Citrix — and comparable in performance, probably faster.
- You do not need to hack XP — it just works.
- You log into the XP box from remote transparently (and I think there is no need to change anything to get a connection, even if authentication is against a domain).
- The NX core technologies are all Open Source and released under the GPL — you can now use a (very inconvenient) commandline at no cost, but you can buy a comfortable (proprietary) NX GUI frontend for money.
- NoMachine are encouraging and offering help to OSS/Free Software implementations for such a frontend too, even if it means competition to them (they have written to this effect even to the LTSP, KDE and GNOME developer mailing lists).

21.4 Network Logon Script Magic

There are several opportunities for creating a custom network startup configuration environment.

- No Logon Script.
- Simple universal Logon Script that applies to all users.
- Use of a conditional Logon Script that applies per user or per group attributes.
- Use of Samba's preexec and postexec functions on access to the NETLOGON share to create a custom logon script and then execute it.
- User of a tool such as KixStart.

The Samba source code tree includes two logon script generation/execution tools. See `examples` directory `genlogon` and `ntlogon` subdirectories.

The following listings are from the `genlogon` directory.

This is the `genlogon.pl` file:

```
#!/usr/bin/perl
#
# genlogon.pl
#
# Perl script to generate user logon scripts on the fly, when users
# connect from a Windows client. This script should be called from
# smb.conf with the %U, %G and %L parameters. I.e:
#
#     root preexec = genlogon.pl %U %G %L
#
# The script generated will perform
# the following:
#
# 1. Log the user connection to /var/log/samba/netlogon.log
# 2. Set the PC's time to the Linux server time (which is maintained
#    daily to the National Institute of Standard's Atomic clock on the
#    internet.
# 3. Connect the user's home drive to H: (H for Home).
# 4. Connect common drives that everyone uses.
# 5. Connect group-specific drives for certain user groups.
# 6. Connect user-specific drives for certain users.
# 7. Connect network printers.

# Log client connection
#($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
#($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
open LOG, ">>/var/log/samba/netlogon.log";
print LOG "$mon/$mday/$year $hour:$min:$sec";
print LOG " - User $ARGV[0] logged into $ARGV[1]\n";
close LOG;

# Start generating logon script
open LOGON, ">/shared/netlogon/$ARGV[0].bat";
print LOGON "@ECHO OFF\r\n";

# Connect shares just use by Software Development group
if ($ARGV[1] eq "SOFTDEV" || $ARGV[0] eq "softdev")
{
    print LOGON "NET USE M: \\$ARGV[2]\SOURCE\r\n";
}
}
```



```

# Connect shares just use by Technical Support staff
if ($ARGV[1] eq "SUPPORT" || $ARGV[0] eq "support")
{
    print LOGON "NET USE S: \\\$ARGV[2]\\SUPPORT\r\n";
}

# Connect shares just used by Administration staff
If ($ARGV[1] eq "ADMIN" || $ARGV[0] eq "admin")
{
    print LOGON "NET USE L: \\\$ARGV[2]\\ADMIN\r\n";
    print LOGON "NET USE K: \\\$ARGV[2]\\MKTING\r\n";
}

# Now connect Printers. We handle just two or three users a little
# differently, because they are the exceptions that have desktop
# printers on LPT1: - all other user's go to the LaserJet on the
# server.
if ($ARGV[0] eq 'jim'
    || $ARGV[0] eq 'yvonne')
{
    print LOGON "NET USE LPT2: \\\$ARGV[2]\\LJET3\r\n";
    print LOGON "NET USE LPT3: \\\$ARGV[2]\\FAXQ\r\n";
}
else
{
    print LOGON "NET USE LPT1: \\\$ARGV[2]\\LJET3\r\n";
    print LOGON "NET USE LPT3: \\\$ARGV[2]\\FAXQ\r\n";
}

# All done! Close the output file.
close LOGON;

```

Those wishing to use more elaborate or capable logon processing system should check out these sites:

- <http://www.craigelachi.e.org/rhacer/ntlogon>
- <http://www.kixtart.org>

21.4.1 Adding Printers without User Intervention

Printers may be added automatically during logon script processing through the use of:

```
C:\> rundll32 printui.dll,PrintUIEntry /?
```

See the documentation in the Microsoft knowledgebase article 189105.²

²<http://support.microsoft.com/default.asp?scid=kb;en-us;189105>

SYSTEM AND ACCOUNT POLICIES

This chapter summarizes the current state of knowledge derived from personal practice and knowledge from Samba mailing list subscribers. Before reproduction of posted information, every effort has been made to validate the information given. Where additional information was uncovered through this validation it is provided also.

22.1 Features and Benefits

When MS Windows NT 3.5 was introduced, the hot new topic was the ability to implement Group Policies for users and groups. Then along came MS Windows NT4 and a few sites started to adopt this capability. How do we know that? By the number of “*boobos*” (or mistakes) administrators made and then requested help to resolve.

By the time that MS Windows 2000 and Active Directory was released, administrators got the message: Group Policies are a good thing! They can help reduce administrative costs and actually make happier users. But adoption of the true potential of MS Windows 200x Active Directory and Group Policy Objects (GPOs) for users and machines were picked up on rather slowly. This was obvious from the Samba mailing list as in 2000 and 2001 when there were few postings regarding GPOs and how to replicate them in a Samba environment.

Judging by the traffic volume since mid 2002, GPOs have become a standard part of the deployment in many sites. This chapter reviews techniques and methods that can be used to exploit opportunities for automation of control over user desktops and network client workstations.

A tool new to Samba — the **editreg** tool — may become an important part of the future Samba administrators’ arsenal is described in this document.

22.2 Creating and Managing System Policies

Under MS Windows platforms, particularly those following the release of MS Windows NT4 and MS Windows 95, it is possible to create a type of file that would be placed in the NETLOGON share of a Domain Controller. As the client logs onto the network, this file

is read and the contents initiate changes to the registry of the client machine. This file allows changes to be made to those parts of the registry that affect users, groups of users, or machines.

For MS Windows 9x/ME, this file must be called `Config.POL` and may be generated using a tool called `poledit.exe`, better known as the Policy Editor. The policy editor was provided on the Windows 98 installation CD, but disappeared again with the introduction of MS Windows Me (Millennium Edition). From comments of MS Windows network administrators, it would appear that this tool became a part of the MS Windows Me Resource Kit.

MS Windows NT4 Server products include the *System Policy Editor* under **Start -> Programs -> Administrative Tools**. For MS Windows NT4 and later clients, this file must be called `NTConfig.POL`.

New with the introduction of MS Windows 2000 was the Microsoft Management Console or MMC. This tool is the new wave in the ever-changing landscape of Microsoft methods for management of network access and security. Every new Microsoft product or technology seems to make the old rules obsolete and introduces newer and more complex tools and methods. To Microsoft's credit, the MMC does appear to be a step forward, but improved functionality comes at a great price.

Before embarking on the configuration of network and system policies, it is highly advisable to read the documentation available from Microsoft's Web site regarding Implementing Profiles and Policies in Windows NT 4.0¹ available from Microsoft. There are a large number of documents in addition to this old one that should also be read and understood. Try searching on the Microsoft Web site for "*Group Policies*".

What follows is a brief discussion with some helpful notes. The information provided here is incomplete — you are warned.

22.2.1 Windows 9x/ME Policies

You need the Windows 98 Group Policy Editor to set up Group Profiles under Windows 9x/ME. It can be found on the original full product Windows 98 installation CD under `tools/reskit/netadmin/poledit`. Install this using the Add/Remove Programs facility and then click on **Have Disk**.

Use the Group Policy Editor to create a policy file that specifies the location of user profiles and/or `My Documents`, and so on. Then save these settings in a file called `Config.POL` that needs to be placed in the root of the `[NETLOGON]` share. If Windows 98 is configured to log onto the Samba Domain, it will automatically read this file and update the Windows 9x/Me registry of the machine as it logs on.

Further details are covered in the Windows 98 Resource Kit documentation.

If you do not take the correct steps, then every so often Windows 9x/ME will check the integrity of the registry and restore its settings from the back-up copy of the registry it stores on each Windows 9x/ME machine. So, you will occasionally notice things changing back to the original settings.

¹http://www.microsoft.com/ntserver/management/deployment/planguide/prof_policies.asp

Install the group policy handler for Windows 9x/Me to pick up Group Policies. Look on the Windows 98 CDROM in `\tools\reskit\netadmin\poledit`. Install group policies on a Windows 9x/Me client by double-clicking on `grouppol.inf`. Log off and on again a couple of times and see if Windows 98 picks up Group Policies. Unfortunately, this needs to be done on every Windows 9x/Me machine that uses Group Policies.

22.2.2 Windows NT4-Style Policy Files

To create or edit `ntconfig.pol` you must use the NT Server Policy Editor, `poledit.exe`, which is included with NT4 Server but not with NT Workstation. There is a Policy Editor on an NT4 Workstation but it is not suitable for creating domain policies. Furthermore, although the Windows 95 Policy Editor can be installed on an NT4 Workstation/Server, it will not work with NT clients. However, the files from the NT Server will run happily enough on an NT4 Workstation.

You need `poledit.exe`, `common.adm` and `winnt.adm`. It is convenient to put the two `*.adm` files in the `c:\winnt\inf` directory, which is where the binary will look for them unless told otherwise. This directory is normally “*hidden*.”

The Windows NT policy editor is also included with the Service Pack 3 (and later) for Windows NT 4.0. Extract the files using `servicpackname /x`, that's `Nt4sp6ai.exe /x` for service pack 6a. The Policy Editor, `poledit.exe`, and the associated template files (`*.adm`) should be extracted as well. It is also possible to download the policy template files for Office97 and get a copy of the Policy Editor. Another possible location is with the Zero Administration Kit available for download from Microsoft.

22.2.2.1 Registry Spoiling

With NT4-style registry-based policy changes, a large number of settings are not automatically reversed as the user logs off. The settings that were in the `NTConfig.POL` file were applied to the client machine registry and apply to the hive key `HKEY_LOCAL_MACHINE` are permanent until explicitly reversed. This is known as tattooing. It can have serious consequences downstream and the administrator must be extremely careful not to lock out the ability to manage the machine at a later date.

22.2.3 MS Windows 200x/XP Professional Policies

Windows NT4 system policies allow the setting of registry parameters specific to users, groups and computers (client workstations) that are members of the NT4-style domain. Such policy files will work with MS Windows 200x/XP clients also.

New to MS Windows 2000, Microsoft recently introduced a style of group policy that confers a superset of capabilities compared with NT4-style policies. Obviously, the tool used to create them is different, and the mechanism for implementing them is much improved.

The older NT4-style registry-based policies are known as *Administrative Templates* in MS Windows 2000/XP Group Policy Objects (GPOs). The later includes the ability to set various security configurations, enforce Internet Explorer browser settings, change and redirect aspects of the users desktop (including the location of `My Documents` files (directory),

as well as intrinsics of where menu items will appear in the Start menu). An additional new feature is the ability to make available particular software Windows applications to particular users and/or groups.

Remember, NT4 policy files are named `NTConfig.POL` and are stored in the root of the `NETLOGON` share on the Domain Controllers. A Windows NT4 user enters a username, password and selects the domain name to which the logon will attempt to take place. During the logon process, the client machine reads the `NTConfig.POL` file from the `NETLOGON` share on the authenticating server and modifies the local registry values according to the settings in this file.

Windows 200x GPOs are feature-rich. They are not stored in the `NETLOGON` share, but rather part of a Windows 200x policy file is stored in the Active Directory itself and the other part is stored in a shared (and replicated) volume called the `SYSVOL` folder. This folder is present on all Active Directory Domain Controllers. The part that is stored in the Active Directory itself is called the Group Policy Container (GPC), and the part that is stored in the replicated share called `SYSVOL` is known as the Group Policy Template (GPT).

With NT4 clients, the policy file is read and executed only as each user logs onto the network. MS Windows 200x policies are much more complex — GPOs are processed and applied at client machine startup (machine specific part) and when the user logs onto the network, the user-specific part is applied. In MS Windows 200x-style policy management, each machine and/or user may be subject to any number of concurrently applicable (and applied) policy sets (GPOs). Active Directory allows the administrator to also set filters over the policy settings. No such equivalent capability exists with NT4-style policy files.

22.2.3.1 Administration of Windows 200x/XP Policies

Instead of using the tool called The System Policy Editor, commonly called `Poedit` (from the executable name `poedit.exe`), GPOs are created and managed using a Microsoft Management Console (MMC) snap-in as follows:

1. Go to the Windows 200x/XP menu **Start->Programs->Administrative Tools** and select the MMC snap-in called **Active Directory Users and Computers**
2. Select the domain or organizational unit (OU) that you wish to manage, then right-click to open the context menu for that object, and select the **Properties**.
3. Left-click on the **Group Policy** tab, then left-click on the **New** tab. Type a name for the new policy you will create.
4. Left-click on the **Edit** tab to commence the steps needed to create the GPO.

All policy configuration options are controlled through the use of policy administrative templates. These files have an `.adm` extension, both in NT4 as well as in Windows 200x/XP. Beware, however, the `.adm` files are not interchangeable across NT4 and Windows 200x. The latter introduces many new features as well as extended definition capabilities. It is well beyond the scope of this documentation to explain how to program `.adm` files; for that the administrator is referred to the Microsoft Windows Resource Kit for your particular version of MS Windows.

NOTE



The MS Windows 2000 Resource Kit contains a tool called `gpolmig.exe`. This tool can be used to migrate an NT4 `NTConfig.POL` file into a Windows 200x style GPO. Be VERY careful how you use this powerful tool. Please refer to the resource kit manuals for specific usage information.

22.3 Managing Account/User Policies

Policies can define a specific user's settings or the settings for a group of users. The resulting policy file contains the registry settings for all users, groups, and computers that will be using the policy file. Separate policy files for each user, group, or computer are not necessary.

If you create a policy that will be automatically downloaded from validating Domain Controllers, you should name the file `NTConfig.POL`. As system administrator, you have the option of renaming the policy file and, by modifying the Windows NT-based workstation, directing the computer to update the policy from a manual path. You can do this by either manually changing the registry or by using the System Policy Editor. This can even be a local path such that each machine has its own policy file, but if a change is necessary to all machines, it must be made individually to each workstation.

When a Windows NT4/200x/XP machine logs onto the network, the client looks in the `NETLOGON` share on the authenticating domain controller for the presence of the `NTConfig.POL` file. If one exists it is downloaded, parsed and then applied to the user's part of the registry.

MS Windows 200x/XP clients that log onto an MS Windows Active Directory security domain may additionally acquire policy settings through Group Policy Objects (GPOs) that are defined and stored in Active Directory itself. The key benefit of using AS GPOs is that they impose no registry *spoiling* effect. This has considerable advantage compared with the use of `NTConfig.POL` (NT4) style policy updates.

In addition to user access controls that may be imposed or applied via system and/or group policies in a manner that works in conjunction with user profiles, the user management environment under MS Windows NT4/200x/XP allows per domain as well as per user account restrictions to be applied. Common restrictions that are frequently used include:

- Logon hours
- Password aging
- Permitted logon from certain machines only
- Account type (local or global)
- User rights

Samba-3.0.0 does not yet implement all account controls that are common to MS Windows NT4/200x/XP. While it is possible to set many controls using the Domain User Manager for MS Windows NT4, only password expiry is functional today. Most of the remaining controls at this time have only stub routines that may eventually be completed to provide actual control. Do not be misled by the fact that a parameter can be set using the NT4 Domain User Manager or in the `NTConfig.POL`.

22.4 Management Tools

Anyone who wishes to create or manage Group Policies will need to be familiar with a number of tools. The following sections describe a few key tools that will help you to create a low maintenance user environment.

22.4.1 Samba Editreg Toolset

A new tool called **editreg** is under development. This tool can be used to edit registry files (called `NTUser.DAT`) that are stored in user and group profiles. `NTConfig.POL` files have the same structure as the `NTUser.DAT` file and can be edited using this tool. **editreg** is being built with the intent to enable `NTConfig.POL` files to be saved in text format and to permit the building of new `NTConfig.POL` files with extended capabilities. It is proving difficult to realize this capability, so do not be surprised if this feature does not materialize. Formal capabilities will be announced at the time that this tool is released for production use.

22.4.2 Windows NT4/200x

The tools that may be used to configure these types of controls from the MS Windows environment are: the NT4 User Manager for Domains, the NT4 System and Group Policy Editor, and the Registry Editor (`regedt32.exe`). Under MS Windows 200x/XP, this is done using the Microsoft Management Console (MMC) with appropriate “*snap-ins*,” the registry editor, and potentially also the NT4 System and Group Policy Editor.

22.4.3 Samba PDC

With a Samba Domain Controller, the new tools for managing user account and policy information include: **smpasswd**, **pdbedit**, **net**, **rpcclient**. The administrator should read the man pages for these tools and become familiar with their use.

22.5 System Startup and Logon Processing Overview

The following attempts to document the order of processing the system and user policies following a system reboot and as part of the user logon:

1. Network starts, then Remote Procedure Call System Service (RPCSS) and Multiple Universal Naming Convention Provider (MUP) start.

2. Where Active Directory is involved, an ordered list of Group Policy Objects (GPOs) is downloaded and applied. The list may include GPOs that:
 - Apply to the location of machines in a Directory.
 - Apply only when settings have changed.
 - Depend on configuration of the scope of applicability: local, site, domain, organizational unit, and so on.No desktop user interface is presented until the above have been processed.
3. Execution of start-up scripts (hidden and synchronous by default).
4. A keyboard action to effect start of logon (Ctrl-Alt-Del).
5. User credentials are validated, user profile is loaded (depends on policy settings).
6. An ordered list of user GPOs is obtained. The list contents depends on what is configured in respect of:
 - Is the user a Domain Member, thus subject to particular policies?
 - Loopback enablement, and the state of the loopback policy (Merge or Replace).
 - Location of the Active Directory itself.
 - Has the list of GPOs changed? No processing is needed if not changed.
7. User Policies are applied from Active Directory. Note: There are several types.
8. Logon scripts are run. New to Windows 200x and Active Directory, logon scripts may be obtained based on Group Policy objects (hidden and executed synchronously). NT4-style logon scripts are then run in a normal window.
9. The User Interface as determined from the GPOs is presented. Note: In a Samba domain (like an NT4 Domain), machine (system) policies are applied at start-up; user policies are applied at logon.

22.6 Common Errors

Policy-related problems can be quite difficult to diagnose and even more difficult to rectify. The following collection demonstrates only basic issues.

22.6.1 Policy Does Not Work

“We have created the `Config.POL` file and put it in the NETLOGON share. It has made no difference to our Win XP Pro machines, they just do not see it. It worked fine with Win 98 but does not work any longer since we upgraded to Win XP Pro. Any hints?”

Policy files are not portable between Windows 9x/Me and MS Windows NT4/200x/XP-based platforms. You need to use the NT4 Group Policy Editor to create a file called `NTConfig.POL` so it is in the correct format for your MS Windows XP Pro clients.

DESKTOP PROFILE MANAGEMENT

23.1 Features and Benefits

Roaming profiles are feared by some, hated by a few, loved by many, and a Godsend for some administrators.

Roaming profiles allow an administrator to make available a consistent user desktop as the user moves from one machine to another. This chapter provides much information regarding how to configure and manage roaming profiles.

While roaming profiles might sound like nirvana to some, they are a real and tangible problem to others. In particular, users of mobile computing tools, where often there may not be a sustained network connection, are often better served by purely local profiles. This chapter provides information to help the Samba administrator deal with those situations.

23.2 Roaming Profiles

WARNING



Roaming profiles support is different for Windows 9x/Me and Windows NT4/200x.

Before discussing how to configure roaming profiles, it is useful to see how Windows 9x/Me and Windows NT4/200x clients implement these features.

Windows 9x/Me clients send a NetUserGetInfo request to the server to get the user's profiles location. However, the response does not have room for a separate profiles location field, only the user's home share. This means that Windows 9x/Me profiles are restricted to being stored in the user's home directory.

Windows NT4/200x clients send a NetSAMLogon RPC request, which contains many fields including a separate field for the location of the user's profiles.

23.2.1 Samba Configuration for Profile Handling

This section documents how to configure Samba for MS Windows client profile support.

23.2.1.1 NT4/200x User Profiles

For example, to support Windows NT4/200x clients, set the following in the [global] section of the `smb.conf` file:

```
logon path = \\profiles\profiles\profilepath\%U\moreprofilepath
```

This is typically implemented like:

```
logon path = \\%L\Profiles\%u
```

where “%L” translates to the name of the Samba server and “%u” translates to the user name.

The default for this option is `\\%N\%U\profile`, namely `\\sambaserver\username\profile`. The `\\%N\%U` service is created automatically by the [homes] service. If you are using a Samba server for the profiles, you must make the share that is specified in the logon path browseable. Please refer to the man page for `smb.conf` in respect of the different semantics of “%L” and “%N”, as well as “%U” and “%u”.

NOTE



MS Windows NT/200x clients at times do not disconnect a connection to a server between logons. It is recommended to not use the *homes* meta-service name as part of the profile share path.

23.2.1.2 Windows 9x/Me User Profiles

To support Windows 9x/Me clients, you must use the `logon home` parameter. Samba has been fixed so `net use /home` now works as well and it, too, relies on the `logon home` parameter.

By using the logon home parameter, you are restricted to putting Windows 9x/Me profiles in the user's home directory. But wait! There is a trick you can use. If you set the following in the [global] section of your `smb.conf` file:

```
logon home = \\%L\%U\profiles
```

then your Windows 9x/Me clients will dutifully put their clients in a subdirectory of your home directory called `.profiles` (making them hidden).

Not only that, but `net use /home` will also work because of a feature in Windows 9x/Me. It removes any directory stuff off the end of the home directory area and only uses the server and share portion. That is, it looks like you specified `\\%L%\%U` for *logon home*.

23.2.1.3 Mixed Windows 9x/Me and Windows NT4/200x User Profiles

You can support profiles for Windows 9x and Windows NT clients by setting both the *logon home* and *logon path* parameters. For example:

```
logon home = \\%L%\%u\.profiles
logon path = \\%L\profiles\%u
```

23.2.1.4 Disabling Roaming Profile Support

A question often asked is: “*How may I enforce use of local profiles?*” or “*How do I disable roaming profiles?*”

There are three ways of doing this:

In `smb.conf` — Affect the following settings and ALL clients will be forced to use a local profile: *logon home* and *logon path*

MS Windows Registry — By using the Microsoft Management Console `gpedit.msc` to instruct your MS Windows XP machine to use only a local profile. This, of course, modifies registry settings. The full path to the option is:

```
Local Computer Policy\
  Computer Configuration\
    Administrative Templates\
      System\
        User Profiles\
```

```
Disable: Only Allow Local User Profiles
```

```
Disable: Prevent Roaming Profile Change from Propagating to the Server
```

Change of Profile Type: — From the start menu right-click on **My Computer icon**, select **Properties**, click on the **User Profiles** tab, select the profile you wish to change from **Roaming** type to **Local**, and click on **Change Type**.

Consult the MS Windows registry guide for your particular MS Windows version for more information about which registry keys to change to enforce use of only local user profiles.

NOTE



The specifics of how to convert a local profile to a roaming profile, or a roaming profile to a local one vary according to the version of MS Windows you are running. Consult the Microsoft MS Windows Resource Kit for your version of Windows for specific information.

23.2.2 Windows Client Profile Configuration Information

23.2.2.1 Windows 9x/Me Profile Setup

When a user first logs in on Windows 9X, the file `user.DAT` is created, as are folders `Start Menu`, `Desktop`, `Programs`, and `Nethood`. These directories and their contents will be merged with the local versions stored in `c:\windows\profiles\username` on subsequent logins, taking the most recent from each. You will need to use the *[global]* options *preserve case* = yes, *short preserve case* = yes and *case sensitive* = no in order to maintain capital letters in shortcuts in any of the profile folders.

The `user.DAT` file contains all the user's preferences. If you wish to enforce a set of preferences, rename their `user.DAT` file to `user.MAN`, and deny them write access to this file.

1. On the Windows 9x/Me machine, go to **Control Panel** -> **Passwords** and select the **User Profiles** tab. Select the required level of roaming preferences. Press **OK**, but do not allow the computer to reboot.
2. On the Windows 9x/Me machine, go to **Control Panel** -> **Network** -> **Client for Microsoft Networks** -> **Preferences**. Select **Log on to NT Domain**. Then, ensure that the Primary Logon is **Client for Microsoft Networks**. Press **OK**, and this time allow the computer to reboot.

Under Windows 9x/ME, profiles are downloaded from the Primary Logon. If you have the Primary Logon as "*Client for Novell Networks*", then the profiles and logon script will be downloaded from your Novell Server. If you have the Primary Logon as "*Windows Logon*", then the profiles will be loaded from the local machine — a bit against the concept of roaming profiles, it would seem!

You will now find that the Microsoft Networks Login box contains `[user, password, domain]` instead of just `[user, password]`. Type in the Samba server's domain name (or any other domain known to exist, but bear in mind that the user will be authenticated against this domain and profiles downloaded from it, if that domain logon server supports it), user name and user's password.

Once the user has been successfully validated, the Windows 9x/Me machine will inform you that `The user has not logged on before` and asks you `Do you wish to save the user's preferences?` Select **Yes**.

Once the Windows 9x/Me client comes up with the desktop, you should be able to examine the contents of the directory specified in the *logon path* on the Samba server and verify that the `Desktop`, `Start Menu`, `Programs` and `Nethood` folders have been created.

These folders will be cached locally on the client, and updated when the user logs off (if you haven't made them read-only by then). You will find that if the user creates further folders or shortcut, that the client will merge the profile contents downloaded with the contents of the profile directory already on the local client, taking the newest folders and shortcut from each set.

If you have made the folders/files read-only on the Samba server, then you will get errors from the Windows 9x/Me machine on logon and logout as it attempts to merge the local and remote profile. Basically, if you have any errors reported by the Windows 9x/Me machine, check the UNIX file permissions and ownership rights on the profile directory contents, on the Samba server.

If you have problems creating user profiles, you can reset the user's local desktop cache, as shown below. When this user next logs in, the user will be told that he/she is logging in "for the first time".

1. Instead of logging in under the [user, password, domain] dialog, press **escape**.
2. Run the **regedit.exe** program, and look in:

```
HKEY_LOCAL_MACHINE\Windows\CurrentVersion\ProfileList
```

You will find an entry for each user of ProfilePath. Note the contents of this key (likely to be `c:\windows\profiles\username`), then delete the key *ProfilePath* for the required user.

3. Exit the registry editor.
4. Search for the user's .PWL password-caching file in the `c:\windows` directory, and delete it.
5. Log off the Windows 9x/Me client.
6. Check the contents of the profile path (see *logon path* described above) and delete the `user.DAT` or `user.MAN` file for the user, making a backup if required.

WARNING



Before deleting the contents of the directory listed in the *ProfilePath* (this is likely to be `c:\windows\profiles\username`), ask the owner if they have any important files stored on their desktop or in their start menu. Delete the contents of the directory *ProfilePath* (making a backup if any of the files are needed).

This will have the effect of removing the local (read-only hidden system file) `user.DAT` in their profile directory, as well as the local "desktop," "nethood," "start menu," and "programs" folders.

If all else fails, increase Samba's debug log levels to between 3 and 10, and/or run a packet sniffer program such as ethereal or **netmon.exe**, and look for error messages.

If you have access to an Windows NT4/200x server, then first set up roaming profiles and/or netlogons on the Windows NT4/200x server. Make a packet trace, or examine the example packet traces provided with Windows NT4/200x server, and see what the differences are with the equivalent Samba trace.

23.2.2.2 Windows NT4 Workstation

When a user first logs in to a Windows NT Workstation, the profile NTuser.DAT is created. The profile location can be now specified through the *logon path* parameter.

There is a parameter that is now available for use with NT Profiles: *logon drive*. This should be set to H: or any other drive, and should be used in conjunction with the new *logon home* parameter.

The entry for the NT4 profile is a directory not a file. The NT help on Profiles mentions that a directory is also created with a .PDS extension. The user, while logging in, must have write permission to create the full profile path (and the folder with the .PDS extension for those situations where it might be created.)

In the profile directory, Windows NT4 creates more folders than Windows 9x/Me. It creates **Application Data** and others, as well as **Desktop**, **Nethood**, **Start Menu**, and **Programs**. The profile itself is stored in a file NTuser.DAT. Nothing appears to be stored in the .PDS directory, and its purpose is currently unknown.

You can use the System Control Panel to copy a local profile onto a Samba server (see NT Help on Profiles; it is also capable of firing up the correct location in the System Control Panel for you). The NT Help file also mentions that renaming NTuser.DAT to NTuser.MAN turns a profile into a mandatory one.

The case of the profile is significant. The file must be called NTuser.DAT or, for a mandatory profile, NTuser.MAN.

23.2.2.3 Windows 2000/XP Professional

You must first convert the profile from a local profile to a domain profile on the MS Windows workstation as follows:

1. Log on as the *local* workstation administrator.
2. Right-click on the **My Computer** Icon, select **Properties**.
3. Click on the **User Profiles** tab.
4. Select the profile you wish to convert (click it once).
5. Click on the **Copy To** button.
6. In the **Permitted to use** box, click on the **Change** button.
7. Click on the **Look in** area that lists the machine name. When you click here, it will open up a selection box. Click on the domain to which the profile must be accessible.

NOTE



You will need to log on if a logon box opens up. For example, connect as *DOMAIN\root*, password: *mypassword*.

8. To make the profile capable of being used by anyone, select “*Everyone*”.
 9. Click on **OK** and the Selection box will close.
 10. Now click on **OK** to create the profile in the path you nominated.
- Done. You now have a profile that can be edited using the Samba **profiles** tool.

NOTE



Under Windows NT/200x, the use of mandatory profiles forces the use of MS Exchange storage of mail data and keeps it out of the desktop profile. That keeps desktop profiles from becoming unusable.

Windows XP Service Pack 1 There is a security check new to Windows XP (or maybe only Windows XP service pack 1). It can be disabled via a group policy in the Active Directory. The policy is called:

```
Computer Configuration\Administrative Templates\System\User Profiles\  
Do not check for user ownership of Roaming Profile Foldersi
```

This should be set to **Enabled**.

Does the new version of Samba have an Active Directory analogue? If so, then you may be able to set the policy through this.

If you cannot set group policies in Samba, then you may be able to set the policy locally on each machine. If you want to try this, then do the following (N.B. I do not know for sure that this will work in the same way as a domain group policy):

1. On the XP workstation, log in with an Administrative account.
2. Click on **Start -> Run**.
3. Type **mmc**.
4. Click on **OK**.
5. A Microsoft Management Console should appear.
6. Click on **File -> Add/Remove Snap-in -> Add**.
7. Double-click on **Group Policy**.

8. Click on **Finish** -> **Close**.
9. Click on **OK**.
10. In the “*Console Root*” window expand **Local Computer Policy** -> **Computer Configuration** -> **Administrative Templates** -> **System** -> **User Profiles**.
11. Double-click on **Do not check for user ownership of Roaming Profile Folders**.
12. Select **Enabled**.
13. Click on **OK**.
14. Close the whole console. You do not need to save the settings (this refers to the console settings rather than the policies you have changed).
15. Reboot.

23.2.3 Sharing Profiles between W9x/Me and NT4/200x/XP Workstations

Sharing of desktop profiles between Windows versions is not recommended. Desktop profiles are an evolving phenomenon and profiles for later versions of MS Windows clients add features that may interfere with earlier versions of MS Windows clients. Probably the more salient reason to not mix profiles is that when logging off an earlier version of MS Windows, the older format of profile contents may overwrite information that belongs to the newer version resulting in loss of profile information content when that user logs on again with the newer version of MS Windows.

If you then want to share the same Start Menu/Desktop with W9x/Me, you will need to specify a common location for the profiles. The `smb.conf` parameters that need to be common are *logon path* and *logon home*.

If you have this set up correctly, you will find separate `user.DAT` and `NTuser.DAT` files in the same profile directory.

23.2.4 Profile Migration from Windows NT4/200x Server to Samba

There is nothing to stop you from specifying any path that you like for the location of users' profiles. Therefore, you could specify that the profile be stored on a Samba server, or any other SMB server, as long as that SMB server supports encrypted passwords.

23.2.4.1 Windows NT4 Profile Management Tools

Unfortunately, the Resource Kit information is specific to the version of MS Windows NT4/200x. The correct resource kit is required for each platform.

Here is a quick guide:

1. On your NT4 Domain Controller, right click on **My Computer**, then select the tab labeled **User Profiles**.
2. Select a user profile you want to migrate and click on it.

NOTE

I am using the term “*migrate*” loosely. You can copy a profile to create a group profile. You can give the user *Everyone* rights to the profile you copy this to. That is what you need to do, since your Samba domain is not a member of a trust relationship with your NT4 PDC.

3. Click on the **Copy To** button.
4. In the box labeled **Copy Profile to** add your new path, e.g., `c:\temp\foobar`
5. Click on **Change** in the **Permitted to use** box.
6. Click on the group “*Everyone*”, click on **OK**. This closes the “*choose user*” box.
7. Now click on **OK**.

Follow the above for every profile you need to migrate.

23.2.4.2 Side Bar Notes

You should obtain the SID of your NT4 domain. You can use `smbpasswd` to do this. Read the man page.

23.2.4.3 `moveuser.exe`

The Windows 200x professional resource kit has `moveuser.exe`. `moveuser.exe` changes the security of a profile from one user to another. This allows the account domain to change, and/or the user name to change.

This command is like the Samba **profiles** tool.

23.2.4.4 Get SID

You can identify the SID by using `GetSID.exe` from the Windows NT Server 4.0 Resource Kit.

Windows NT 4.0 stores the local profile information in the registry under the following key: `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList`

Under the `ProfileList` key, there will be subkeys named with the SIDs of the users who have logged on to this computer. (To find the profile information for the user whose locally cached profile you want to move, find the SID for the user with the `GetSID.exe` utility.) Inside the appropriate user’s subkey, you will see a string value named `ProfileImagePath`.

23.3 Mandatory Profiles

A Mandatory Profile is a profile that the user does not have the ability to overwrite. During the user's session, it may be possible to change the desktop environment, however, as the user logs out all changes made will be lost. If it is desired to not allow the user any ability to change the desktop environment, then this must be done through policy settings. See the previous chapter.

NOTE



Under NO circumstances should the profile directory (or its contents) be made read-only as this may render the profile un-usable. Where it is essential to make a profile read-only within the UNIX file system, this can be done but then you absolutely must use the **fake-permissions** VFS module to instruct MS Windows NT/200x/XP clients that the Profile has write permission for the user. See Section 19.3.3.

For MS Windows NT4/200x/XP, the above method can also be used to create mandatory profiles. To convert a group profile into a mandatory profile, simply locate the `NTUser.DAT` file in the copied profile and rename it to `NTUser.MAN`.

For MS Windows 9x/ME, it is the `User.DAT` file that must be renamed to `User.MAN` to effect a mandatory profile.

23.4 Creating and Managing Group Profiles

Most organizations are arranged into departments. There is a nice benefit in this fact since usually most users in a department require the same desktop applications and the same desktop layout. MS Windows NT4/200x/XP will allow the use of Group Profiles. A Group Profile is a profile that is created first using a template (example) user. Then using the profile migration tool (see above), the profile is assigned access rights for the user group that needs to be given access to the group profile.

The next step is rather important. Instead of assigning a group profile to users (Using User Manager) on a “*per user*” basis, the group itself is assigned the now modified profile.

NOTE



Be careful with Group Profiles. If the user who is a member of a group also has a personal profile, then the result will be a fusion (merge) of the two.

23.5 Default Profile for Windows Users

MS Windows 9x/Me and NT4/200x/XP will use a default profile for any user for whom a profile does not already exist. Armed with a knowledge of where the default profile is located on the Windows workstation, and knowing which registry keys effect the path from which the default profile is created, it is possible to modify the default profile to one that has been optimized for the site. This has significant administrative advantages.

23.5.1 MS Windows 9x/Me

To enable default per use profiles in Windows 9x/ME, you can either use the Windows 98 System Policy Editor or change the registry directly.

To enable default per user profiles in Windows 9x/ME, launch the System Policy Editor, then select **File -> Open Registry**, next click on the **Local Computer** icon, click on **Windows 98 System**, select **User Profiles**, and click on the enable box. Remember to save the registry changes.

To modify the registry directly, launch the Registry Editor (**regedit.exe**) and select the hive `HKEY_LOCAL_MACHINE\Network\Logon`. Now add a DWORD type key with the name “*User Profiles*,” to enable user profiles to set the value to 1; to disable user profiles set it to 0.

23.5.1.1 User Profile Handling with Windows 9x/Me

When a user logs on to a Windows 9x/Me machine, the local profile path, `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\ProfileList`, is checked for an existing entry for that user.

If the user has an entry in this registry location, Windows 9x/Me checks for a locally cached version of the user profile. Windows 9x/Me also checks the user’s home directory (or other specified directory if the location has been modified) on the server for the User Profile. If a profile exists in both locations, the newer of the two is used. If the User Profile exists on the server, but does not exist on the local machine, the profile on the server is downloaded and used. If the User Profile only exists on the local machine, that copy is used.

If a User Profile is not found in either location, the Default User Profile from the Windows 9x/Me machine is used and copied to a newly created folder for the logged on user. At log off, any changes that the user made are written to the user’s local profile. If the user has a roaming profile, the changes are written to the user’s profile on the server.

23.5.2 MS Windows NT4 Workstation

On MS Windows NT4, the default user profile is obtained from the location `%System-Root%\Profiles` which in a default installation will translate to `C:\Windows NT\Profiles`. Under this directory on a clean install there will be three (3) directories: **Administrator**, **All Users**, and **Default User**.

The **All Users** directory contains menu settings that are common across all system users. The **Default User** directory contains menu entries that are customizable per user depending on the profile settings chosen/created.

When a new user first logs onto an MS Windows NT4 machine, a new profile is created from:

- All Users settings.
- Default User settings (contains the default `NTuser.DAT` file).

When a user logs onto an MS Windows NT4 machine that is a member of a Microsoft security domain, the following steps are followed in respect of profile handling:

1. The users' account information that is obtained during the logon process contains the location of the users' desktop profile. The profile path may be local to the machine or it may be located on a network share. If there exists a profile at the location of the path from the user account, then this profile is copied to the location `%SystemRoot%\Profiles\%USERNAME%`. This profile then inherits the settings in the **All Users** profile in the `%SystemRoot%\Profiles` location.
2. If the user account has a profile path, but at its location a profile does not exist, then a new profile is created in the `%SystemRoot%\Profiles\%USERNAME%` directory from reading the **Default User** profile.
3. If the **NETLOGON** share on the authenticating server (logon server) contains a policy file (`NTConfig.POL`), then its contents are applied to the `NTuser.DAT` which is applied to the `HKEY_CURRENT_USER` part of the registry.
4. When the user logs out, if the profile is set to be a roaming profile it will be written out to the location of the profile. The `NTuser.DAT` file is then recreated from the contents of the `HKEY_CURRENT_USER` contents. Thus, should there not exist in the **NETLOGON** share an `NTConfig.POL` at the next logon, the effect of the previous `NTConfig.POL` will still be held in the profile. The effect of this is known as tattooing.

MS Windows NT4 profiles may be *local* or *roaming*. A local profile will be stored in the `%SystemRoot%\Profiles\%USERNAME%` location. A roaming profile will also remain stored in the same way, unless the following registry key is created as shown:

```
HKEY_LOCAL_MACHINE\SYSTEM\Software\Microsoft\Windows NT\CurrentVersion\  
winlogon\DeleteRoamingCache=dword:0000000
```

In this case, the local copy (in `%SystemRoot%\Profiles\%USERNAME%`) will be deleted on logout.

Under MS Windows NT4, default locations for common resources like **My Documents** may be redirected to a network share by modifying the following registry keys. These changes may be affected via use of the System Policy Editor. To do so may require that you create your own template extension for the policy editor to allow this to be done through the GUI. Another way to do this is by way of first creating a default user profile, then while logged in as that user, run **regedt32** to edit the key settings.

The Registry Hive key that affects the behavior of folders that are part of the default user profile are controlled by entries on Windows NT4 is:

```
HKEY_CURRENT_USER
  \Software
    \Microsoft
      \Windows
        \CurrentVersion
          \Explorer
            \User Shell Folders
```

The above hive key contains a list of automatically managed folders. The default entries are shown in Table 23.1.

Table 23.1. User Shell Folder Registry Keys Default Values

Name	Default Value
AppData	%USERPROFILE%\Application Data
Desktop	%USERPROFILE%\Desktop
Favorites	%USERPROFILE%\Favorites
NetHood	%USERPROFILE%\NetHood
PrintHood	%USERPROFILE%\PrintHood
Programs	%USERPROFILE%\Start Menu\Programs
Recent	%USERPROFILE%\Recent
SendTo	%USERPROFILE%\SendTo
Start Menu	%USERPROFILE%\Start Menu
Startup	%USERPROFILE%\Start Menu\Programs\Startup

The registry key that contains the location of the default profile settings is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\
User Shell Folders
```

The default entries are shown in Table 23.2.

Table 23.2. Defaults of Profile Settings Registry Keys

Common Desktop	%SystemRoot%\Profiles\All Users\Desktop
Common Programs	%SystemRoot%\Profiles\All Users\Programs
Common Start Menu	%SystemRoot%\Profiles\All Users\Start Menu
Common Startup	%SystemRoot%\Profiles\All Users\Start Menu\Programs\Startup

23.5.3 MS Windows 200x/XP

NOTE



MS Windows XP Home Edition does use default per user profiles, but cannot participate in domain security, cannot log onto an NT/ADS-style domain, and thus can obtain the profile only from itself. While there are benefits in doing this, the beauty of those MS Windows clients that can participate in domain logon processes allows the administrator to create a global default profile and enforce it through the use of Group Policy Objects (GPOs).

When a new user first logs onto an MS Windows 200x/XP machine, the default profile is obtained from `C:\Documents and Settings\Default User`. The administrator can modify or change the contents of this location and MS Windows 200x/XP will gladly use it. This is far from the optimum arrangement since it will involve copying a new default profile to every MS Windows 200x/XP client workstation.

When MS Windows 200x/XP participates in a domain security context, and if the default user profile is not found, then the client will search for a default profile in the NETLOGON share of the authenticating server. In MS Windows parlance, `%LOGONSERVER%\NETLOGON\Default User`, and if one exists there it will copy this to the workstation to the `C:\Documents and Settings\` under the Windows login name of the user.

NOTE



This path translates, in Samba parlance, to the `smb.conf [NETLOGON]` share. The directory should be created at the root of this share and must be called `Default Profile`.

If a default profile does not exist in this location, then MS Windows 200x/XP will use the local default profile.

On logging out, the users' desktop profile will be stored to the location specified in the registry settings that pertain to the user. If no specific policies have been created or passed to the client during the login process (as Samba does automatically), then the user's profile will be written to the local machine only under the path `C:\Documents and Settings\%USERNAME%`.

Those wishing to modify the default behavior can do so through these three methods:

- Modify the registry keys on the local machine manually and place the new default profile in the NETLOGON share root. This is not recommended as it is maintenance intensive.

- Create an NT4-style NTConfig.POL file that specified this behavior and locate this file in the root of the NETLOGON share along with the new default profile.
- Create a GPO that enforces this through Active Directory, and place the new default profile in the NETLOGON share.

The registry hive key that effects the behavior of folders that are part of the default user profile are controlled by entries on Windows 200x/XP is:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders\

The above hive key contains a list of automatically managed folders. The default entries are shown in Table 23.3

Table 23.3. Defaults of Default User Profile Paths Registry Keys

Name	Default Value
AppData	%USERPROFILE%\Application Data
Cache	%USERPROFILE%\Local Settings\Temporary Internet Files
Cookies	%USERPROFILE%\Cookies
Desktop	%USERPROFILE%\Desktop
Favorites	%USERPROFILE%\Favorites
History	%USERPROFILE%\Local Settings\History
Local AppData	%USERPROFILE%\Local Settings\Application Data
Local Settings	%USERPROFILE%\Local Settings
My Pictures	%USERPROFILE%\My Documents\My Pictures
NetHood	%USERPROFILE%\NetHood
Personal	%USERPROFILE%\My Documents
PrintHood	%USERPROFILE%\PrintHood
Programs	%USERPROFILE%\Start Menu\Programs
Recent	%USERPROFILE%\Recent
SendTo	%USERPROFILE%\SendTo
Start Menu	%USERPROFILE%\Start Menu
Startup	%USERPROFILE%\Start Menu\Programs\Startup
Templates	%USERPROFILE%\Templates

There is also an entry called “*Default*” that has no value set. The default entry is of type REG_SZ, all the others are of type REG_EXPAND_SZ.

It makes a huge difference to the speed of handling roaming user profiles if all the folders are stored on a dedicated location on a network server. This means that it will not be necessary to write the Outlook PST file over the network for every login and logout.

To set this to a network location, you could use the following examples:

```
%LOGONSERVER%\%USERNAME%\Default Folders
```

This would store the folders in the user’s home directory under a directory called **Default Folders**. You could also use:

```
\\SambaServer\FolderShare\%USERNAME%
```

in which case the default folders will be stored in the server named *SambaServer* in the share called *FolderShare* under a directory that has the name of the MS Windows user as seen by the Linux/UNIX file system.

Please note that once you have created a default profile share, you MUST migrate a user's profile (default or custom) to it.

MS Windows 200x/XP profiles may be *Local* or *Roaming*. A roaming profile will be cached locally unless the following registry key is created:

```
HKEY_LOCAL_MACHINE\SYSTEM\Software\Microsoft\Windows NT\CurrentVersion\  
winlogon\DeleteRoamingCache=dword:00000001
```

In this case, the local cache copy will be deleted on logout.

23.6 Common Errors

The following are some typical errors, problems and questions that have been asked on the Samba mailing lists.

23.6.1 Configuring Roaming Profiles for a Few Users or Groups

With Samba-2.2.x, the choice you have is to enable or disable roaming profiles support. It is a global only setting. The default is to have roaming profiles and the default path will locate them in the user's home directory.

If disabled globally, then no one will have roaming profile ability. If enabled and you want it to apply only to certain machines, then on those machines on which roaming profile support is not wanted it is then necessary to disable roaming profile handling in the registry of each such machine.

With Samba-3, you can have a global profile setting in `smb.conf` and you can override this by per-user settings using the Domain User Manager (as with MS Windows NT4/ Win 200xx).

In any case, you can configure only one profile per user. That profile can be either:

- A profile unique to that user.
- A mandatory profile (one the user cannot change).
- A group profile (really should be mandatory, that is unchangable).

23.6.2 Cannot Use Roaming Profiles

A user requested the following: *“I do not want Roaming profiles to be implemented. I want to give users a local profile alone. Please help me, I am totally lost with this error. For the past two days I tried everything, I googled around but found no useful pointers. Please help me.”*

The choices are:

Local profiles — I know of no registry keys that will allow auto-deletion of LOCAL profiles on log out.

Roaming profiles — As a user logs onto the network, a centrally stored profile is copied to the workstation to form a local profile. This local profile will persist (remain on the workstation disk) unless a registry key is changed that will cause this profile to be automatically deleted on logout.

The roaming profile choices are:

Personal roaming profiles — These are typically stored in a profile share on a central (or conveniently located local) server.

Workstations cache (store) a local copy of the profile. This cached copy is used when the profile cannot be downloaded at next logon.

Group profiles — These are loaded from a central profile server.

Mandatory profiles — Mandatory profiles can be created for a user as well as for any group that a user is a member of. Mandatory profiles cannot be changed by ordinary users. Only the administrator can change or reconfigure a mandatory profile.

A Windows NT4/200x/XP profile can vary in size from 130KB to very large. Outlook PST files are most often part of the profile and can be many GB in size. On average (in a well controlled environment), roaming profile size of 2MB is a good rule of thumb to use for planning purposes. In an undisciplined environment, I have seen up to 2GB profiles. Users tend to complain when it takes an hour to log onto a workstation but they harvest the fruits of folly (and ignorance).

The point of all the above is to show that roaming profiles and good controls of how they can be changed as well as good discipline make up for a problem-free site.

Microsoft's answer to the PST problem is to store all email in an MS Exchange Server backend. This removes the need for a PST file.

Local profiles mean:

- If each machine is used by many users, then much local disk storage is needed for local profiles.
- Every workstation the user logs into has its own profile; these can be very different from machine to machine.

On the other hand, use of roaming profiles means:

- The network administrator can control the desktop environment of all users.
- Use of mandatory profiles drastically reduces network management overheads.
- In the long run, users will experience fewer problems.

23.6.3 Changing the Default Profile

“When the client logs onto the Domain Controller, it searches for a profile to download. Where do I put this default profile?”

First, the Samba server needs to be configured as a Domain Controller. This can be done by setting in `smb.conf`:

```
security = user  
os level = 32 (or more)  
domain logons = Yes
```

There must be a `[netlogon]` share that is world readable. It is a good idea to add a logon script to pre-set printer and drive connections. There is also a facility for automatically synchronizing the workstation time clock with that of the logon server (another good thing to do).

NOTE



To invoke auto-deletion of roaming profile from the local workstation cache (disk storage), use the Group Policy Editor to create a file called `NTConfig.POL` with the appropriate entries. This file needs to be located in the `netlogon` share root directory.

Windows clients need to be members of the domain. Workgroup machines do not use network logons so they do not interoperate with domain profiles.

For roaming profiles, add to `smb.conf`:

```
logon path = \\%N\profiles\%U  
logon drive = H:
```

PAM-BASED DISTRIBUTED AUTHENTICATION

This chapter should help you to deploy Winbind-based authentication on any PAM-enabled UNIX/Linux system. Winbind can be used to enable User-Level application access authentication from any MS Windows NT Domain, MS Windows 200x Active Directory-based domain, or any Samba-based domain environment. It will also help you to configure PAM-based local host access controls that are appropriate to your Samba configuration.

In addition to knowing how to configure Winbind into PAM, you will learn generic PAM management possibilities and in particular how to deploy tools like `pam_smbpass.so` to your advantage.

NOTE



The use of Winbind requires more than PAM configuration alone. Please refer to Chapter 20, *Winbind: Use of Domain Accounts*, for further information regarding Winbind.

24.1 Features and Benefits

A number of UNIX systems (e.g., Sun Solaris), as well as the xxxxBSD family and Linux, now utilize the Pluggable Authentication Modules (PAM) facility to provide all authentication, authorization and resource control services. Prior to the introduction of PAM, a decision to use an alternative to the system password database (`/etc/passwd`) would require the provision of alternatives for all programs that provide security services. Such a choice would involve provision of alternatives to programs such as: **login**, **passwd**, **chown**, and so on.

PAM provides a mechanism that disconnects these security programs from the underlying authentication/authorization infrastructure. PAM is configured by making appropriate

modifications to one file `/etc/pam.conf` (Solaris), or by editing individual control files that are located in `/etc/pam.d`.

On PAM-enabled UNIX/Linux systems, it is an easy matter to configure the system to use any authentication backend so long as the appropriate dynamically loadable library modules are available for it. The backend may be local to the system, or may be centralized on a remote server.

PAM support modules are available for:

/etc/passwd — There are several PAM modules that interact with this standard UNIX user database. The most common are called: `pam_unix.so`, `pam_unix2.so`, `pam_pwdb.so` and `pam_userdb.so`.

Kerberos — The `pam_krb5.so` module allows the use of any Kerberos compliant server. This tool is used to access MIT Kerberos, Heimdal Kerberos, and potentially Microsoft Active Directory (if enabled).

LDAP — The `pam_ldap.so` module allows the use of any LDAP v2 or v3 compatible backend server. Commonly used LDAP backend servers include: OpenLDAP v2.0 and v2.1, Sun ONE iIdentity server, Novell eDirectory server, Microsoft Active Directory.

NetWare Bindery — The `pam_ncp_auth.so` module allows authentication off any bindery-enabled NetWare Core Protocol-based server.

SMB Password — This module, called `pam_smbpass.so`, will allow user authentication off the `passwd` backend that is configured in the Samba `smb.conf` file.

SMB Server — The `pam_smb_auth.so` module is the original MS Windows networking authentication tool. This module has been somewhat outdated by the `Winbind` module.

Winbind — The `pam_winbind.so` module allows Samba to obtain authentication from any MS Windows Domain Controller. It can just as easily be used to authenticate users for access to any PAM-enabled application.

RADIUS — There is a PAM RADIUS (Remote Access Dial-In User Service) authentication module. In most cases, administrators will need to locate the source code for this tool and compile and install it themselves. RADIUS protocols are used by many routers and terminal servers.

Of the above, Samba provides the `pam_smbpasswd.so` and the `pam_winbind.so` modules alone.

Once configured, these permit a remarkable level of flexibility in the location and use of distributed Samba Domain Controllers that can provide wide area network bandwidth effi-

cient authentication services for PAM-capable systems. In effect, this allows the deployment of centrally managed and maintained distributed authentication from a single-user account database.

24.2 Technical Discussion

PAM is designed to provide the system administrator with a great deal of flexibility in configuration of the privilege granting applications of their system. The local configuration of system security controlled by PAM is contained in one of two places: either the single system file, `/etc/pam.conf`, or the `/etc/pam.d/` directory.

24.2.1 PAM Configuration Syntax

In this section we discuss the correct syntax of and generic options respected by entries to these files. PAM-specific tokens in the configuration file are case insensitive. The module paths, however, are case sensitive since they indicate a file's name and reflect the case dependence of typical file systems. The case-sensitivity of the arguments to any given module is defined for each module in turn.

In addition to the lines described below, there are two special characters provided for the convenience of the system administrator: comments are preceded by a “`#`” and extend to the next end-of-line; also, module specification lines may be extended with a “`\`” escaped newline.

If the PAM authentication module (loadable link library file) is located in the default location, then it is not necessary to specify the path. In the case of Linux, the default location is `/lib/security`. If the module is located outside the default, then the path must be specified as:

```
auth required /other_path/pam_strange_module.so
```

24.2.1.1 Anatomy of `/etc/pam.d` Entries

The remaining information in this subsection was taken from the documentation of the Linux-PAM project. For more information on PAM, see The Official Linux-PAM home page.¹

A general configuration line of the `/etc/pam.conf` file has the following form:

```
service-name  module-type  control-flag  module-path  args
```

Below, we explain the meaning of each of these tokens. The second (and more recently adopted) way of configuring Linux-PAM is via the contents of the `/etc/pam.d/` directory. Once we have explained the meaning of the above tokens, we will describe this method.

¹<http://ftp.kernel.org/pub/linux/libs/pam/>

service-name — The name of the service associated with this entry. Frequently, the service name is the conventional name of the given application. For example, **ftpd**, **rlogind** and **su**, and so on.

There is a special service-name reserved for defining a default authentication mechanism. It has the name *OTHER* and may be specified in either lower- or upper-case characters. Note, when there is a module specified for a named service, the *OTHER* entries are ignored.

module-type — One of (currently) four types of module. The four types are as follows:

- **auth**: This module type provides two aspects of authenticating the user. It establishes that the user is who he claims to be by instructing the application to prompt the user for a password or other means of identification. Secondly, the module can grant group membership (independently of the `/etc/groups` file discussed above) or other privileges through its credential granting properties.
- **account**: This module performs non-authentication-based account management. It is typically used to restrict/permit access to a service based on the time of day, currently available system resources (maximum number of users) or perhaps the location of the applicant user “*root*” login only on the console.
- **session**: Primarily, this module is associated with doing things that need to be done for the user before and after they can be given service. Such things include the logging of information concerning the opening and closing of some data exchange with a user, mounting directories, and so on.
- **password**: This last module type is required for updating the authentication token associated with the user. Typically, there is one module for each “*challenge/response*” -based authentication (*auth*) module type.

control-flag — The control-flag is used to indicate how the PAM library will react to the success or failure of the module it is associated with. Since modules can be stacked (modules of the same type execute in series, one after another), the control-flags determine the relative importance of each module. The application is not made aware of the individual success or failure of modules listed in the `/etc/pam.conf` file. Instead, it receives a summary success or fail response from the Linux-PAM library. The order of execution of these modules is that of the entries in the `/etc/pam.conf` file; earlier entries are executed before later ones. As of Linux-PAM v0.60, this control-flag can be defined with one of two syntaxes.

The simpler (and historical) syntax for the control-flag is a single keyword defined to indicate the severity of concern associated with the success or failure of a specific module. There are four such keywords: *required*, *requisite*, *sufficient* and *optional*.

The Linux-PAM library interprets these keywords in the following manner:

- **required**: This indicates that the success of the module is required for the

module-type facility to succeed. Failure of this module will not be apparent to the user until all of the remaining modules (of the same module-type) have been executed.

- *requisite*: Like required, however, in the case that such a module returns a failure, control is directly returned to the application. The return value is that associated with the first required or requisite module to fail. This flag can be used to protect against the possibility of a user getting the opportunity to enter a password over an unsafe medium. It is conceivable that such behavior might inform an attacker of valid accounts on a system. This possibility should be weighed against the not insignificant concerns of exposing a sensitive password in a hostile environment.
- *sufficient*: The success of this module is deemed *sufficient* to satisfy the Linux-PAM library that this module-type has succeeded in its purpose. In the event that no previous required module has failed, no more “*stacked*” modules of this type are invoked. (In this case, subsequent required modules are not invoked). A failure of this module is not deemed as fatal to satisfying the application that this module-type has succeeded.
- *optional*: As its name suggests, this control-flag marks the module as not being critical to the success or failure of the user’s application for service. In general, Linux-PAM ignores such a module when determining if the module stack will succeed or fail. However, in the absence of any definite successes or failures of previous or subsequent stacked modules, this module will determine the nature of the response to the application. One example of this latter case, is when the other modules return something like PAM_IGNORE.

The more elaborate (newer) syntax is much more specific and gives the administrator a great deal of control over how the user is authenticated. This form of the control flag is delimited with square brackets and consists of a series of *value=action* tokens:

```
[value1=action1 value2=action2 ...]
```

Here, *value1* is one of the following return values:

```
success; open_err; symbol_err; service_err; system_err; buf_err;  
perm_denied; auth_err; cred_insufficient; authinfo_unavail;  
user_unknown; maxtries; new_authtok_reqd; acct_expired; session_err;  
cred_unavail; cred_expired; cred_err; no_module_data; conv_err;  
authtok_err; authtok_recover_err; authtok_lock_busy;  
authtok_disable_aging; try_again; ignore; abort; authtok_expired;  
module_unknown; bad_item; and default.
```

The last of these (*default*) can be used to set the action for those return values that are not explicitly defined.

The *action1* can be a positive integer or one of the following tokens: *ignore*; *ok*;

done; *bad*; *die*; and *reset*. A positive integer, *J*, when specified as the action, can be used to indicate that the next *J* modules of the current module-type will be skipped. In this way, the administrator can develop a moderately sophisticated stack of modules with a number of different paths of execution. Which path is taken can be determined by the reactions of individual modules.

- *ignore*: When used with a stack of modules, the module's return status will not contribute to the return code the application obtains.
- *bad*: This action indicates that the return code should be thought of as indicative of the module failing. If this module is the first in the stack to fail, its status value will be used for that of the whole stack.
- *die*: Equivalent to *bad* with the side effect of terminating the module stack and PAM immediately returning to the application.
- *ok*: This tells PAM that the administrator thinks this return code should contribute directly to the return code of the full stack of modules. In other words, if the former state of the stack would lead to a return of PAM_SUCCESS, the module's return code will override this value. Note, if the former state of the stack holds some value that is indicative of a modules failure, this *ok* value will not be used to override that value.
- *done*: Equivalent to *ok* with the side effect of terminating the module stack and PAM immediately returning to the application.
- *reset*: Clears all memory of the state of the module stack and starts again with the next stacked module.

Each of the four keywords: *required*; *requisite*; *sufficient*; and *optional*, have an equivalent expression in terms of the [...] syntax. They are as follows:

- *required* is equivalent to `[success=ok new_authtok_reqd=ok ignore=ignore default=bad]`.
- *requisite* is equivalent to `[success=ok new_authtok_reqd=ok ignore=ignore default=die]`.
- *sufficient* is equivalent to `[success=done new_authtok_reqd=done default=ignore]`.
- *optional* is equivalent to `[success=ok new_authtok_reqd=ok default=ignore]`.

Just to get a feel for the power of this new syntax, here is a taste of what you can do with it. With Linux-PAM-0.63, the notion of client plug-in agents was introduced. This is something that makes it possible for PAM to support machine-machine authentication using the transport protocol inherent to the client/server application. With the `[... value=action ...]` control syntax, it is possible for an application to be configured to support binary prompts with compliant clients, but to gracefully fall over into an alternative authentication mode for older, legacy applications.

module-path — The path-name of the dynamically loadable object file; the pluggable module itself. If the first character of the module path is “/”, it is assumed to be

a complete path. If this is not the case, the given module path is appended to the default module path: `/lib/security` (but see the notes above).

The arguments are a list of tokens that are passed to the module when it is invoked, much like arguments to a typical Linux shell command. Generally, valid arguments are optional and are specific to any given module. Invalid arguments are ignored by a module, however, when encountering an invalid argument, the module is required to write an error to `syslog(3)`. For a list of generic options, see the next section.

If you wish to include spaces in an argument, you should surround that argument with square brackets. For example:

```
squid auth required pam_mysql.so user=passwd_query passwd=mada \
db=eminence [query=select user_name from internet_service where \
user_name=%u and password=PASSWORD(%p) and service=web_proxy]
```

When using this convention, you can include “`’`” characters inside the string, and if you wish to have a “`”` character inside the string that will survive the argument parsing, you should use “`\”`”. In other words:

```
[..[..\]\..] --> ..[...]
```

Any line in one of the configuration files that is not formatted correctly will generally tend (erring on the side of caution) to make the authentication process fail. A corresponding error is written to the system log files with a call to `syslog(3)`.

24.2.2 Example System Configurations

The following is an example `/etc/pam.d/login` configuration file. This example had all options uncommented and is probably not usable because it stacks many conditions before allowing successful completion of the login process. Essentially all conditions can be disabled by commenting them out, except the calls to `pam_pwdb.so`.

24.2.2.1 PAM: Original Login Config

```
##%PAM-1.0
# The PAM configuration file for the login service
#
auth          required      pam_securetty.so
auth          required      pam_nologin.so
# auth        required      pam_dialup.so
# auth        optional      pam_mail.so
auth          required      pam_pwdb.so shadow md5
# account     requisite     pam_time.so
account       required      pam_pwdb.so
```

```

session      required    pam_pwdb.so
# session    optional    pam_lastlog.so
# password   required    pam_cracklib.so retry=3
password     required    pam_pwdb.so shadow md5

```

24.2.2.2 PAM: Login Using pam_smbpass

PAM allows use of replaceable modules. Those available on a sample system include:

```
$/bin/ls /lib/security
```

```

pam_access.so      pam_ftp.so         pam_limits.so
pam_ncp_auth.so    pam_rhosts_auth.so pam_stress.so
pam_cracklib.so    pam_group.so       pam_listfile.so
pam_nologin.so     pam_rootok.so      pam_tally.so
pam_deny.so        pam_issue.so       pam_mail.so
pam_permit.so      pam_securetty.so   pam_time.so
pam_dialup.so      pam_lastlog.so     pam_mkhome.so
pam_pwdb.so        pam_shells.so      pam_unix.so
pam_env.so         pam_ldap.so        pam_motd.so
pam_radius.so      pam_smbpass.so     pam_unix_acct.so
pam_wheel.so       pam_unix_auth.so   pam_unix_passwd.so
pam_userdb.so      pam_warn.so        pam_unix_session.so

```

The following example for the login program replaces the use of the `pam_pwdb.so` module that uses the system password database (`/etc/passwd`, `/etc/shadow`, `/etc/group`) with the module `pam_smbpass.so`, which uses the Samba database which contains the Microsoft MD4 encrypted password hashes. This database is stored in either `/usr/local/samba/private/smbpasswd`, `/etc/samba/smbpasswd`, or in `/etc/samba.d/smbpasswd`, depending on the Samba implementation for your UNIX/Linux system. The `pam_smbpass.so` module is provided by Samba version 2.2.1 or later. It can be compiled by specifying the `--with-pam_smbpass` options when running Samba's `configure` script. For more information on the `pam_smbpass` module, see the documentation in the `source/pam_smbpass` directory of the Samba source distribution.

```

#%PAM-1.0
# The PAM configuration file for the login service
#
auth      required    pam_smbpass.so nodelay
account   required    pam_smbpass.so nodelay
session   required    pam_smbpass.so nodelay
password  required    pam_smbpass.so nodelay

```

The following is the PAM configuration file for a particular Linux system. The default condition uses `pam_pwdb.so`.

```

#%PAM-1.0
# The PAM configuration file for the samba service
#
auth      required      pam_pwdb.so nullok nodelay shadow audit
account   required      pam_pwdb.so audit nodelay
session   required      pam_pwdb.so nodelay
password  required      pam_pwdb.so shadow md5

```

In the following example, the decision has been made to use the **smbpasswd** database even for basic Samba authentication. Such a decision could also be made for the **passwd** program and would thus allow the **smbpasswd** passwords to be changed using the **passwd** program:

```

#%PAM-1.0
# The PAM configuration file for the samba service
#
auth      required      pam_smbpass.so nodelay
account   required      pam_pwdb.so audit nodelay
session   required      pam_pwdb.so nodelay
password  required      pam_smbpass.so nodelay smbconf=/etc/samba.d/smb.conf

```

NOTE



PAM allows stacking of authentication mechanisms. It is also possible to pass information obtained within one PAM module through to the next module in the PAM stack. Please refer to the documentation for your particular system implementation for details regarding the specific capabilities of PAM in this environment. Some Linux implementations also provide the `pam_stack.so` module that allows all authentication to be configured in a single central file. The `pam_stack.so` method has some devoted followers on the basis that it allows for easier administration. As with all issues in life though, every decision makes trade-offs, so you may want to examine the PAM documentation for further helpful information.

24.2.3 smb.conf PAM Configuration

There is an option in `smb.conf` called `obey pam restrictions`. The following is from the online help for this option in SWAT;

When Samba is configured to enable PAM support (i.e., `--with-pam`), this parameter will control whether or not Samba should obey PAM's account and session management directives. The default behavior is to use PAM for cleartext authentication only and to ignore

any account or session management. Samba always ignores PAM for authentication in the case of *encrypt passwords* = yes. The reason is that PAM modules cannot support the challenge/response authentication mechanism needed in the presence of SMB password encryption.

Default: *obey pam restrictions* = no

24.2.4 Remote CIFS Authentication Using `winbindd.so`

All operating systems depend on the provision of users credentials acceptable to the platform. UNIX requires the provision of a user identifier (UID) as well as a group identifier (GID). These are both simple integer type numbers that are obtained from a password backend such as `/etc/passwd`.

Users and groups on a Windows NT server are assigned a relative ID (RID) which is unique for the domain when the user or group is created. To convert the Windows NT user or group into a UNIX user or group, a mapping between RIDs and UNIX user and group IDs is required. This is one of the jobs that `winbind` performs.

As Winbind users and groups are resolved from a server, user and group IDs are allocated from a specified range. This is done on a first come, first served basis, although all existing users and groups will be mapped as soon as a client performs a user or group enumeration command. The allocated UNIX IDs are stored in a database file under the Samba lock directory and will be remembered.

The astute administrator will realize from this that the combination of `pam_smbpass.so`, `winbindd` and a distributed *passwd backend*, such as `ldap`, will allow the establishment of a centrally managed, distributed user/password database that can also be used by all PAM-aware (e.g., Linux) programs and applications. This arrangement can have particularly potent advantages compared with the use of Microsoft Active Directory Service (ADS) in so far as the reduction of wide area network authentication traffic.

WARNING



The RID to UNIX ID database is the only location where the user and group mappings are stored by `winbindd`. If this file is deleted or corrupted, there is no way for `winbindd` to determine which user and group IDs correspond to Windows NT user and group RIDs.

24.2.5 Password Synchronization Using `pam_smbpass.so`

`pam_smbpass` is a PAM module that can be used on conforming systems to keep the `smbpasswd` (Samba password) database in sync with the UNIX password file. PAM (Pluggable Authentication Modules) is an API supported under some UNIX operating systems, such as Solaris, HP-UX and Linux, that provides a generic interface to authentication mechanisms.

This module authenticates a local `smbpasswd` user database. If you require support for authenticating against a remote SMB server, or if you are concerned about the presence of SUID root binaries on your system, it is recommended that you use `pam_winbind` instead.

Options recognized by this module are shown in Table 24.1.

Table 24.1. Options recognized by `pam_smbpass`

<code>debug</code>	log more debugging info.
<code>audit</code>	like <code>debug</code> , but also logs unknown usernames.
<code>use_first_pass</code>	do not prompt the user for passwords; take them from PAM_ items instead.
<code>try_first_pass</code>	try to get the password from a previous PAM module fall back to prompting the user.
<code>use_authtok</code>	like <code>try_first_pass</code> , but <code>*fail*</code> if the new PAM_AUTHTOK has not been previously set (intended for stacking password modules only).
<code>not_set_pass</code>	do not make passwords used by this module available to other modules.
<code>nodelay</code>	do not insert ~1 second delays on authentication failure.
<code>nullok</code>	null passwords are allowed.
<code>nonull</code>	null passwords are not allowed. Used to override the Samba configuration.
<code>migrate</code>	only meaningful in an “ <i>auth</i> ” context; used to update <code>smbpasswd</code> file with a password used for successful authentication.
<code>smbconf=</code> <i>file</i>	specify an alternate path to the <code>smb.conf</code> file.

The following are examples of the use of `pam_smbpass.so` in the format of Linux `/etc/pam.d/` files structure. Those wishing to implement this tool on other platforms will need to adapt this appropriately.

24.2.5.1 Password Synchronization Configuration

A sample PAM configuration that shows the use of `pam_smbpass` to make sure `private/smbpasswd` is kept in sync when `/etc/passwd` (`/etc/shadow`) is changed. Useful when an expired password might be changed by an application (such as `ssh`).

```

#%PAM-1.0
# password-sync
#
auth      requisite    pam_nologin.so
auth      required     pam_unix.so
account   required     pam_unix.so
password  requisite     pam_cracklib.so retry=3
password  requisite     pam_unix.so shadow md5 use_authtok try_first_pass
password  required     pam_smbpass.so nullok use_authtok try_first_pass
session   required     pam_unix.so

```

24.2.5.2 Password Migration Configuration

A sample PAM configuration that shows the use of `pam_smbpass` to migrate from plaintext to encrypted passwords for Samba. Unlike other methods, this can be used for users who have never connected to Samba shares: password migration takes place when users `ftp` in, login using `ssh`, pop their mail, and so on.

```

#%PAM-1.0
# password-migration
#
auth      requisite    pam_nologin.so
# pam_smbpass is called IF pam_unix succeeds.
auth      requisite    pam_unix.so
auth      optional     pam_smbpass.so migrate
account   required     pam_unix.so
password  requisite     pam_cracklib.so retry=3
password  requisite     pam_unix.so shadow md5 use_authtok try_first_pass
password  optional     pam_smbpass.so nullok use_authtok try_first_pass
session   required     pam_unix.so

```

24.2.5.3 Mature Password Configuration

A sample PAM configuration for a mature `smbpasswd` installation. `private/smbpasswd` is fully populated, and we consider it an error if the SMB password does not exist or does not match the UNIX password.

```

#%PAM-1.0
# password-mature
#
auth      requisite    pam_nologin.so
auth      required     pam_unix.so
account   required     pam_unix.so
password  requisite     pam_cracklib.so retry=3
password  requisite     pam_unix.so shadow md5 use_authtok try_first_pass
password  required     pam_smbpass.so use_authtok use_first_pass
session   required     pam_unix.so

```

24.2.5.4 Kerberos Password Integration Configuration

A sample PAM configuration that shows `pam_smbpass` used together with `pam_krb5`. This could be useful on a Samba PDC that is also a member of a Kerberos realm.

```

#%PAM-1.0
# kdc-pdc

```



```
#
auth      requisite    pam_nologin.so
auth      requisite    pam_krb5.so
auth      optional     pam_smbpass.so migrate
account   required     pam_krb5.so
password  requisite    pam_cracklib.so retry=3
password  optional     pam_smbpass.so nullok use_authok try_first_pass
password  required     pam_krb5.so use_authok try_first_pass
session   required     pam_krb5.so
```

24.3 Common Errors

PAM can be fickle and sensitive to configuration glitches. Here we look at a few cases from the Samba mailing list.

24.3.1 pam_winbind Problem

A user reported: I have the following PAM configuration:

```
auth required /lib/security/pam_securetty.so
auth sufficient /lib/security/pam_winbind.so
auth sufficient /lib/security/pam_unix.so use_first_pass nullok
auth required /lib/security/pam_stack.so service=system-auth
auth required /lib/security/pam_nologin.so
account required /lib/security/pam_stack.so service=system-auth
account required /lib/security/pam_winbind.so
password required /lib/security/pam_stack.so service=system-auth
```

When I open a new console with [ctrl][alt][F1], I can't log in with my user "pitie". I have tried with user "scienceu+pitie" also.

Answer: The problem may lie with your inclusion of `pam_stack.so service=system-auth`. That file often contains a lot of stuff that may duplicate what you are already doing. Try commenting out the `pam_stack` lines for `auth` and `account` and see if things work. If they do, look at `/etc/pam.d/system-auth` and copy only what you need from it into your `/etc/pam.d/login` file. Alternately, if you want all services to use Winbind, you can put the Winbind-specific stuff in `/etc/pam.d/system-auth`.

24.3.2 Winbind Is Not Resolving Users and Groups

"My smb.conf file is correctly configured. I have specified idmap uid = 12000, and idmap gid = 3000-3500 and winbind is running. When I do the following it all works fine."

```
root# wbinfo -u
```

```
MIDEARTH+maryo
MIDEARTH+jackb
MIDEARTH+ameds
...
MIDEARTH+root
```

```
root# wbinfo -g
MIDEARTH+Domain Users
MIDEARTH+Domain Admins
MIDEARTH+Domain Guests
...
MIDEARTH+Accounts
```

```
root# getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
...
maryo:x:15000:15003:Mary Orville:/home/MIDEARTH/maryo:/bin/false
```

“But this command fails.”

```
root# chown maryo a_file
chown: 'maryo': invalid user
```

“This is driving me nuts! What can be wrong?”

Answer: Your system is likely running **nscd**, the name service caching daemon. Shut it down, do not restart it! You will find your problem resolved.

INTEGRATING MS WINDOWS NETWORKS WITH SAMBA

This section deals with NetBIOS over TCP/IP name to IP address resolution. If your MS Windows clients are not configured to use NetBIOS over TCP/IP, then this section does not apply to your installation. If your installation involves the use of NetBIOS over TCP/IP then this section may help you to resolve networking problems.

NOTE



NetBIOS over TCP/IP has nothing to do with NetBEUI. NetBEUI is NetBIOS over Logical Link Control (LLC). On modern networks it is highly advised to not run NetBEUI at all. Note also there is no such thing as NetBEUI over TCP/IP — the existence of such a protocol is a complete and utter misapprehension.

25.1 Features and Benefits

Many MS Windows network administrators have never been exposed to basic TCP/IP networking as it is implemented in a UNIX/Linux operating system. Likewise, many UNIX and Linux administrators have not been exposed to the intricacies of MS Windows TCP/IP-based networking (and may have no desire to be either).

This chapter gives a short introduction to the basics of how a name can be resolved to its IP address for each operating system environment.

25.2 Background Information

Since the introduction of MS Windows 2000, it is possible to run MS Windows networking without the use of NetBIOS over TCP/IP. NetBIOS over TCP/IP uses UDP port 137 for NetBIOS name resolution and uses TCP port 139 for NetBIOS session services. When

NetBIOS over TCP/IP is disabled on MS Windows 2000 and later clients, then only the TCP port 445 will be used and the UDP port 137 and TCP port 139 will not.

NOTE



When using Windows 2000 or later clients, if NetBIOS over TCP/IP is not disabled, then the client will use UDP port 137 (NetBIOS Name Service, also known as the Windows Internet Name Service or WINS), TCP port 139 and TCP port 445 (for actual file and print traffic).

When NetBIOS over TCP/IP is disabled, the use of DNS is essential. Most installations that disable NetBIOS over TCP/IP today use MS Active Directory Service (ADS). ADS requires Dynamic DNS with Service Resource Records (SRV RR) and with Incremental Zone Transfers (IXFR). Use of DHCP with ADS is recommended as a further means of maintaining central control over the client workstation network configuration.

25.3 Name Resolution in a Pure UNIX/Linux World

The key configuration files covered in this section are:

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/host.conf`
- `/etc/nsswitch.conf`

25.3.1 `/etc/hosts`

This file contains a static list of IP addresses and names.

```
127.0.0.1    localhost localhost.localdomain
192.168.1.1 bigbox.kenya.org bigbox    alias4box
```

The purpose of `/etc/hosts` is to provide a name resolution mechanism so users do not need to remember IP addresses.

Network packets that are sent over the physical network transport layer communicate not via IP addresses but rather using the Media Access Control address, or MAC address. IP addresses are currently 32 bits in length and are typically presented as four (4) decimal numbers that are separated by a dot (or period). For example, 168.192.1.1.

MAC Addresses use 48 bits (or 6 bytes) and are typically represented as two-digit hexadecimal numbers separated by colons: 40:8e:0a:12:34:56.

Every network interface must have a MAC address. Associated with a MAC address may be one or more IP addresses. There is no relationship between an IP address and a MAC address; all such assignments are arbitrary or discretionary in nature. At the most basic level, all network communications take place using MAC addressing. Since MAC addresses must be globally unique and generally remain fixed for any particular interface, the assignment of an IP address makes sense from a network management perspective. More than one IP address can be assigned per MAC address. One address must be the primary IP address — this is the address that will be returned in the ARP reply.

When a user or a process wants to communicate with another machine, the protocol implementation ensures that the “*machine name*” or “*host name*” is resolved to an IP address in a manner that is controlled by the TCP/IP configuration control files. The file `/etc/hosts` is one such file.

When the IP address of the destination interface has been determined, a protocol called ARP/RARP is used to identify the MAC address of the target interface. ARP stands for Address Resolution Protocol and is a broadcast-oriented method that uses User Datagram Protocol (UDP) to send a request to all interfaces on the local network segment using the all 1s MAC address. Network interfaces are programmed to respond to two MAC addresses only; their own unique address and the address `ff:ff:ff:ff:ff:ff`. The reply packet from an ARP request will contain the MAC address and the primary IP address for each interface.

The `/etc/hosts` file is foundational to all UNIX/Linux TCP/IP installations and as a minimum will contain the localhost and local network interface IP addresses and the primary names by which they are known within the local machine. This file helps to prime the pump so a basic level of name resolution can exist before any other method of name resolution becomes available.

25.3.2 `/etc/resolv.conf`

This file tells the name resolution libraries:

- The name of the domain to which the machine belongs.
- The name(s) of any domains that should be automatically searched when trying to resolve unqualified host names to their IP address.
- The name or IP address of available Domain Name Servers that may be asked to perform name-to-address translation lookups.

25.3.3 `/etc/host.conf`

`/etc/host.conf` is the primary means by which the setting in `/etc/resolv.conf` may be effected. It is a critical configuration file. This file controls the order by which name resolution may proceed. The typical structure is:

```
order hosts,bind
multi on
```

then both addresses should be returned. Please refer to the man page for `host.conf` for further details.

25.3.4 `/etc/nsswitch.conf`

This file controls the actual name resolution targets. The file typically has resolver object specifications as follows:

```
# /etc/nsswitch.conf
#
# Name Service Switch configuration file.
#

passwd:      compat
# Alternative entries for password authentication are:
# passwd:    compat files nis ldap winbind
shadow:     compat
group:      compat

hosts:      files nis dns
# Alternative entries for host name resolution are:
# hosts:    files dns nis nis+ hesiod db compat ldap wins
networks:   nis files dns

ethers:     nis files
protocols:  nis files
rpc:        nis files
services:   nis files
```

Of course, each of these mechanisms requires that the appropriate facilities and/or services are correctly configured.

It should be noted that unless a network request/message must be sent, TCP/IP networks are silent. All TCP/IP communications assume a principal of speaking only when necessary.

Starting with version 2.2.0, Samba has Linux support for extensions to the name service switch infrastructure so Linux clients will be able to obtain resolution of MS Windows NetBIOS names to IP Addresses. To gain this functionality, Samba needs to be compiled with appropriate arguments to the make command (i.e., `make nsswitch/libnss_wins.so`). The resulting library should then be installed in the `/lib` directory and the `wins` parameter needs to be added to the `"hosts:"` line in the `/etc/nsswitch.conf` file. At this point, it will be possible to ping any MS Windows machine by its NetBIOS machine name, as long as that machine is within the workgroup to which both the Samba machine and the MS Windows machine belong.

25.4 Name Resolution as Used within MS Windows Networking

MS Windows networking is predicated about the name each machine is given. This name is known variously (and inconsistently) as the “*computer name*,” “*machine name*,” “*net-working name*,” “*netbios name*,” or “*SMB name*.” All terms mean the same thing with the exception of “*netbios name*” that can also apply to the name of the workgroup or the domain name. The terms “*workgroup*” and “*domain*” are really just a simple name with which the machine is associated. All NetBIOS names are exactly 16 characters in length. The 16th character is reserved. It is used to store a one-byte value that indicates service level information for the NetBIOS name that is registered. A NetBIOS machine name is, therefore, registered for each service type that is provided by the client/server.

Table 25.1 and Table 25.2 list typical NetBIOS name/service type registrations.

Table 25.1. Unique NetBIOS Names

MACHINENAME<00>	Server Service is running on MACHINENAME
MACHINENAME<03>	Generic Machine Name (NetBIOS name)
MACHINENAME<20>	LanMan Server service is running on MACHINENAME
WORKGROUP<1b>	Domain Master Browser

Table 25.2. Group Names

WORKGROUP<03>	Generic Name registered by all members of WORKGROUP
WORKGROUP<1c>	Domain Controllers / Netlogon Servers
WORKGROUP<1d>	Local Master Browsers
WORKGROUP<1e>	Browser Election Service

It should be noted that all NetBIOS machines register their own names as per the above. This is in vast contrast to TCP/IP installations where traditionally the system administrator will determine in the `/etc/hosts` or in the DNS database what names are associated with each IP address.

One further point of clarification should be noted. The `/etc/hosts` file and the DNS records do not provide the NetBIOS name type information that MS Windows clients depend on to locate the type of service that may be needed. An example of this is what happens when an MS Windows client wants to locate a domain logon server. It finds this service and the IP address of a server that provides it by performing a lookup (via a NetBIOS broadcast) for enumeration of all machines that have registered the name type `*<1c>`. A logon request is then sent to each IP address that is returned in the enumerated list of IP addresses. Whichever machine first replies, it then ends up providing the logon services.

The name “*workgroup*” or “*domain*” really can be confusing since these have the added significance of indicating what is the security architecture of the MS Windows network. The term “*workgroup*” indicates that the primary nature of the network environment is that of a peer-to-peer design. In a WORKGROUP, all machines are responsible for their own security, and generally such security is limited to the use of just a password (known as Share Level security). In most situations with peer-to-peer networking, the users who control their own machines will simply opt to have no security at all. It is possible to have

User Level Security in a WORKGROUP environment, thus requiring the use of a user name and a matching password.

MS Windows networking is thus predetermined to use machine names for all local and remote machine message passing. The protocol used is called Server Message Block (SMB) and this is implemented using the NetBIOS protocol (Network Basic Input Output System). NetBIOS can be encapsulated using LLC (Logical Link Control) protocol — in which case the resulting protocol is called NetBEUI (Network Basic Extended User Interface). NetBIOS can also be run over IPX (Internetworking Packet Exchange) protocol as used by Novell NetWare, and it can be run over TCP/IP protocols — in which case the resulting protocol is called NBT or NetBT, the NetBIOS over TCP/IP.

MS Windows machines use a complex array of name resolution mechanisms. Since we are primarily concerned with TCP/IP, this demonstration is limited to this area.

25.4.1 The NetBIOS Name Cache

All MS Windows machines employ an in-memory buffer in which is stored the NetBIOS names and IP addresses for all external machines that machine has communicated with over the past 10-15 minutes. It is more efficient to obtain an IP address for a machine from the local cache than it is to go through all the configured name resolution mechanisms.

If a machine whose name is in the local name cache has been shut down before the name had been expired and flushed from the cache, then an attempt to exchange a message with that machine will be subject to time-out delays. Its name is in the cache, so a name resolution lookup will succeed, but the machine cannot respond. This can be frustrating for users but is a characteristic of the protocol.

The MS Windows utility that allows examination of the NetBIOS name cache is called “*nbtstat*”. The Samba equivalent of this is called **nmblookup**.

25.4.2 The LMHOSTS File

This file is usually located in MS Windows NT 4.0 or Windows 200x/XP in the directory C:\WINNT\SYSTEM32\DRIVERS\ETC and contains the IP Address and the machine name in matched pairs. The LMHOSTS file performs NetBIOS name to IP address mapping.

It typically looks like this:

```
# Copyright (c) 1998 Microsoft Corp.
#
# This is a sample LMHOSTS file used by the Microsoft Wins Client (NetBIOS
# over TCP/IP) stack for Windows98
#
# This file contains the mappings of IP addresses to NT computernames
# (NetBIOS) names. Each entry should be kept on an individual line.
# The IP address should be placed in the first column followed by the
# corresponding computername. The address and the computername
# should be separated by at least one space or tab. The "#" character
```



```
# is generally used to denote the start of a comment (see the exceptions
# below).
#
# This file is compatible with Microsoft LAN Manager 2.x TCP/IP lmhosts
# files and offers the following extensions:
#
#     #PRE
#     #DOM:<domain>
#     #INCLUDE <filename>
#     #BEGIN_ALTERNATE
#     #END_ALTERNATE
#     \Oxnn (non-printing character support)
#
# Following any entry in the file with the characters "#PRE" will cause
# the entry to be preloaded into the name cache. By default, entries are
# not preloaded, but are parsed only after dynamic name resolution fails.
#
# Following an entry with the "#DOM:<domain>" tag will associate the
# entry with the domain specified by <domain>. This effects how the
# browser and logon services behave in TCP/IP environments. To preload
# the host name associated with #DOM entry, it is necessary to also add a
# #PRE to the line. The <domain> is always preloaded although it will not
# be shown when the name cache is viewed.
#
# Specifying "#INCLUDE <filename>" will force the RFC NetBIOS (NBT)
# software to seek the specified <filename> and parse it as if it were
# local. <filename> is generally a UNC-based name, allowing a
# centralized lmhosts file to be maintained on a server.
# It is ALWAYS necessary to provide a mapping for the IP address of the
# server prior to the #INCLUDE. This mapping must use the #PRE directive.
# In addition the share "public" in the example below must be in the
# LanManServer list of "NullSessionShares" in order for client machines to
# be able to read the lmhosts file successfully. This key is under
# \machine\system\currentcontrolset\services\lanmanserver\
# parameters>nullsessionshares
# in the registry. Simply add "public" to the list found there.
#
# The #BEGIN_ and #END_ALTERNATE keywords allow multiple #INCLUDE
# statements to be grouped together. Any single successful include
# will cause the group to succeed.
#
# Finally, non-printing characters can be embedded in mappings by
# first surrounding the NetBIOS name in quotations, then using the
# \Oxnn notation to specify a hex value for a non-printing character.
#
# The following example illustrates all of these extensions:
#
# 102.54.94.97      rhino      #PRE #DOM:networking #net group's DC
```

```
# 102.54.94.102    "appname  \0x14"      #special app server
# 102.54.94.123    popular  #PRE          #source server
# 102.54.94.117    localsrv #PRE          #needed for the include
#
# #BEGIN_ALTERNATE
# #INCLUDE \\localsrv\public\lmhosts
# #INCLUDE \\rhino\public\lmhosts
# #END_ALTERNATE
#
# In the above example, the "appname" server contains a special
# character in its name, the "popular" and "localsrv" server names are
# preloaded, and the "rhino" server name is specified so it can be used
# to later #INCLUDE a centrally maintained lmhosts file if the "localsrv"
# system is unavailable.
#
# Note that the whole file is parsed including comments on each lookup,
# so keeping the number of comments to a minimum will improve performance.
# Therefore it is not advisable to simply add lmhosts file entries onto the
# end of this file.
```

25.4.3 HOSTS File

This file is usually located in MS Windows NT 4.0 or Windows 200x/XP in the directory C:\WINNT\SYSTEM32\DRIVERS\ETC and contains the IP Address and the IP hostname in matched pairs. It can be used by the name resolution infrastructure in MS Windows, depending on how the TCP/IP environment is configured. This file is in every way the equivalent of the UNIX/Linux `/etc/hosts` file.

25.4.4 DNS Lookup

This capability is configured in the TCP/IP setup area in the network configuration facility. If enabled, an elaborate name resolution sequence is followed, the precise nature of which is dependant on how the NetBIOS Node Type parameter is configured. A Node Type of 0 means that NetBIOS broadcast (over UDP broadcast) is used if the name that is the subject of a name lookup is not found in the NetBIOS name cache. If that fails then DNS, HOSTS and LMHOSTS are checked. If set to Node Type 8, then a NetBIOS Unicast (over UDP Unicast) is sent to the WINS Server to obtain a lookup before DNS, HOSTS, LMHOSTS, or broadcast lookup is used.

25.4.5 WINS Lookup

A WINS (Windows Internet Name Server) service is the equivalent of the rfc1001/1002 specified NBNS (NetBIOS Name Server). A WINS server stores the names and IP addresses that are registered by a Windows client if the TCP/IP setup has been given at least one WINS Server IP Address.

To configure Samba to be a WINS server, the following parameter needs to be added to the `smb.conf` file:

```
wins support = Yes
```

To configure Samba to use a WINS server, the following parameters are needed in the `smb.conf` file:

```
wins support = No  
wins server = xxx.xxx.xxx.xxx
```

where `xxx.xxx.xxx.xxx` is the IP address of the WINS server.

For information about setting up Samba as a WINS server, read Chapter 9, *Network Browsing*.

25.5 Common Errors

TCP/IP network configuration problems find every network administrator sooner or later. The cause can be anything from keyboard mishaps, forgetfulness, simple mistakes, and carelessness. Of course, no one is ever deliberately careless!

25.5.1 Pinging Works Only in One Way

“I can ping my Samba server from Windows, but I cannot ping my Windows machine from the Samba server.”

Answer: The Windows machine was at IP Address 192.168.1.2 with netmask 255.255.255.0, the Samba server (Linux) was at IP Address 192.168.1.130 with netmask 255.255.255.128. The machines were on a local network with no external connections.

Due to inconsistent netmasks, the Windows machine was on network 192.168.1.0/24, while the Samba server was on network 192.168.1.128/25 — logically a different network.

25.5.2 Very Slow Network Connections

A common cause of slow network response includes:

- Client is configured to use DNS and the DNS server is down.
- Client is configured to use remote DNS server, but the remote connection is down.
- Client is configured to use a WINS server, but there is no WINS server.
- Client is not configured to use a WINS server, but there is a WINS server.
- Firewall is filtering our DNS or WINS traffic.

25.5.3 Samba Server Name Change Problem

“The name of the Samba server was changed, Samba was restarted, Samba server cannot be pinged by new name from MS Windows NT4 Workstation, but it does still respond to ping using the old name. Why?”

From this description, three things are obvious:

- WINS is not in use, only broadcast-based name resolution is used.
- The Samba server was renamed and restarted within the last 10-15 minutes.
- The old Samba server name is still in the NetBIOS name cache on the MS Windows NT4 Workstation.

To find what names are present in the NetBIOS name cache on the MS Windows NT4 machine, open a **cmd** shell and then:

```
C:\> nbtstat -n
```

NetBIOS Local Name Table

Name	Type	Status
FRODO	<03> UNIQUE	Registered
ADMINISTRATOR	<03> UNIQUE	Registered
FRODO	<00> UNIQUE	Registered
SARDON	<00> GROUP	Registered
FRODO	<20> UNIQUE	Registered
FRODO	<1F> UNIQUE	Registered

```
C:\> nbtstat -c
```

NetBIOS Remote Cache Name Table

Name	Type	Host Address	Life [sec]
GANDALF <20>	UNIQUE	192.168.1.1	240

```
C:\>
```

In the above example, GANDALF is the Samba server and FRODO is the MS Windows NT4 Workstation. The first listing shows the contents of the Local Name Table (i.e., Identity information on the MS Windows workstation) and the second shows the NetBIOS name in the NetBIOS name cache. The name cache contains the remote machines known to this workstation.

UNICODE/CHARSETS

26.1 Features and Benefits

Every industry eventually matures. One of the great areas of maturation is in the focus that has been given over the past decade to make it possible for anyone anywhere to use a computer. It has not always been that way, in fact, not so long ago it was common for software to be written for exclusive use in the country of origin.

Of all the effort that has been brought to bear on providing native language support for all computer users, the efforts of the Openi18n organization¹ is deserving of special mention.

Samba-2.x supported a single locale through a mechanism called *codepages*. Samba-3 is destined to become a truly trans-global file and printer-sharing platform.

26.2 What Are Charsets and Unicode?

Computers communicate in numbers. In texts, each number will be translated to a corresponding letter. The meaning that will be assigned to a certain number depends on the *character set (charset)* that is used.

A charset can be seen as a table that is used to translate numbers to letters. Not all computers use the same charset (there are charsets with German umlauts, Japanese characters, and so on). The American Standard Code for Information Interchange (ASCII) encoding system has been the normative character encoding scheme used by computers to date. This employs a charset that contains 256 characters. Using this mode of encoding each character takes exactly one byte.

There are also charsets that support extended characters, but those need at least twice as much storage space as does ASCII encoding. Such charsets can contain **256 * 256 = 65536** characters, which is more than all possible characters one could think of. They are called multibyte charsets because they use more than one byte to store one character.

One standardized multibyte charset encoding scheme is known as unicode². A big advantage of using a multibyte charset is that you only need one. There is no need to make sure two

¹<http://www.openi18n.org/>

²<http://www.unicode.org/>

computers use the same charset when they are communicating.

Old Windows clients use single-byte charsets, named *codepages*, by Microsoft. However, there is no support for negotiating the charset to be used in the SMB/CIFS protocol. Thus, you have to make sure you are using the same charset when talking to an older client. Newer clients (Windows NT, 200x, XP) talk unicode over the wire.

26.3 Samba and Charsets

As of Samba-3, Samba can (and will) talk unicode over the wire. Internally, Samba knows of three kinds of character sets:

unix charset — This is the charset used internally by your operating system. The default is UTF-8, which is fine for most systems, which covers all characters in all languages. The default in previous Samba releases was ASCII.

display charset — This is the charset Samba will use to print messages on your screen. It should generally be the same as the *unix charset*.

dos charset — This is the charset Samba uses when communicating with DOS and Windows 9x/Me clients. It will talk unicode to all newer clients. The default depends on the charsets you have installed on your system. Run `testparm -v | grep "dos charset"` to see what the default is on your system.

26.4 Conversion from Old Names

Because previous Samba versions did not do any charset conversion, characters in filenames are usually not correct in the UNIX charset but only for the local charset used by the DOS/Windows clients.

Bjoern Jacke has written a utility named `convmv`³ that can convert whole directory structures to different charsets with one single command.

26.5 Common Errors

26.5.1 CP850.so Can't Be Found

"Samba is complaining about a missing CP850.so file."

Answer: CP850 is the default *dos charset*. The *dos charset* is used to convert data to the codepage used by your dos clients. If you do not have any dos clients, you can safely ignore this message.

³<http://j3e.de/linux/convmv/>

CP850 should be supported by your local iconv implementation. Make sure you have all the required packages installed. If you compiled Samba from source, make sure to configure found iconv.

BACKUP TECHNIQUES

27.1 Features and Benefits

The Samba project is over ten years old. During the early history of Samba, UNIX administrators were its key implementors. UNIX administrators will use UNIX system tools to backup UNIX system files. Over the past four years, an increasing number of Microsoft network administrators have taken an interest in Samba. This is reflected in the questions about backup in general on the Samba mailing lists.

27.2 Discussion of Backup Solutions

During discussions at a Microsoft Windows training course, one of the pro-UNIX delegates stunned the class when he pointed out that Windows NT4 is so limiting compared with UNIX. He likened UNIX to a mechano set that has an unlimited number of tools that are simple, efficient, and, in combination, capable of achieving any desired outcome.

One of the Windows networking advocates retorted that if she wanted a mechano set, she would buy one. She made it clear that a complex single tool that does more than is needed but does it with a clear purpose and intent is preferred by some like her.

Please note that all information here is provided as is and without recommendation of fitness or suitability. The network administrator is strongly encouraged to perform due-diligence research before implementing any backup solution, whether free software or commercial.

A useful Web site I recently stumbled across that you might like to refer to is located at www.allmerchants.com.

The following three free software projects might also merit consideration.

27.2.1 BackupPC

BackupPC version 2.0.0 has been released on SourceForge.¹ New features include support for **rsync/rsyncd** and internationalization of the CGI interface (including English, French, Spanish, and German).

¹<http://backuppc.sourceforge.net>

BackupPC is a high-performance Perl-based package for backing up Linux, UNIX or Windows PCs and laptops to a server's disk. BackupPC is highly configurable and easy to install and maintain. SMB (via smbclient), **tar** over **rsh/ssh** or **rsync/rsyncd** are used to extract client data.

Given the ever decreasing cost of disks and raid systems, it is now practical and cost effective to backup a large number of machines onto a server's local disk or network storage. This is what BackupPC does.

Key features are pooling of identical files (big savings in server disk space), compression, and a comprehensive CGI interface that allows users to browse backups and restore files.

BackupPC is free software distributed under a GNU GPL license. BackupPC runs on Linux/UNIX/freenix servers, and has been tested on Linux, UNIX, Windows 9x/ME, Windows 98, Windows 200x, Windows XP, and Mac OSX clients.

27.2.2 Rsync

rsync is a flexible program for efficiently copying files or directory trees.

rsync has many options to select which files will be copied and how they are to be transferred. It may be used as an alternative to **ftp**, **http**, **scp**, or **rcp**.

The rsync remote-update protocol allows rsync to transfer just the differences between two sets of files across the network link, using an efficient checksum-search algorithm described in the technical report that accompanies the rsync package.

Some of the additional features of rsync are:

- Support for copying links, devices, owners, groups, and permissions.
- Exclude and exclude-from options are similar to GNU tar.
- A CVS exclude mode for ignoring the same files that CVS would ignore.
- Can use any transparent remote shell, including rsh or ssh.
- Does not require root privileges.
- Pipelining of file transfers to minimize latency costs.
- Support for anonymous or authenticated rsync servers (ideal for mirroring).

27.2.3 Amanda

Amanda, the Advanced Maryland Automatic Network Disk Archiver, is a backup system that allows the administrator of a LAN to set up a single master backup server to back up multiple hosts to a single large capacity tape drive. Amanda uses native dump and/or GNU tar facilities and can back up a large number of workstations running multiple versions of UNIX. Recent versions can also use Samba to back up Microsoft Windows hosts.

For more information regarding Amanda, please check the www.amanda.org/ site.²

²<http://www.amanda.org/>

27.2.4 BOBS: Browseable Online Backup System

Browseable Online Backup System (BOBS) is a complete online backup system. Uses large disks for storing backups and lets users browse the files using a Web browser. Handles some special files like AppleDouble and icon files.

The home page for BOBS is located at bobs.sourceforge.net.³

³<http://bobs.sourceforge.net/>

HIGH AVAILABILITY

28.1 Features and Benefits

Network administrators are often concerned about the availability of file and print services. Network users are inclined toward intolerance of the services they depend on to perform vital task responsibilities.

A sign in a computer room served to remind staff of their responsibilities. It read:

All humans fail, in both great and small ways we fail continually. Machines fail too. Computers are machines that are managed by humans, the fallout from failure can be spectacular. Your responsibility is to deal with failure, to anticipate it and to eliminate it as far as is humanly and economically wise to achieve. Are your actions part of the problem or part of the solution?

If we are to deal with failure in a planned and productive manner, then first we must understand the problem. That is the purpose of this chapter.

Parenthetically, in the following discussion there are seeds of information on how to provision a network infrastructure against failure. Our purpose here is not to provide a lengthy dissertation on the subject of high availability. Additionally, we have made a conscious decision to not provide detailed working examples of high availability solutions; instead we present an overview of the issues in the hope that someone will rise to the challenge of providing a detailed document that is focused purely on presentation of the current state of knowledge and practice in high availability as it applies to the deployment of Samba and other CIFS/SMB technologies.

28.2 Technical Discussion

The following summary was part of a presentation by Jeremy Allison at the SambaXP 2003 conference that was held at Goettingen, Germany, in April 2003. Material has been added from other sources, but it was Jeremy who inspired the structure that follows.

28.2.1 The Ultimate Goal

All clustering technologies aim to achieve one or more of the following:

- Obtain the maximum affordable computational power.
- Obtain faster program execution.
- Deliver unstopable services.
- Avert points of failure.
- Exact most effective utilization of resources.

A clustered file server ideally has the following properties:

- All clients can connect transparently to any server.
- A server can fail and clients are transparently reconnected to another server.
- All servers server out the same set of files.
- All file changes are immediately seen on all servers.
 - Requires a distributed file system.
- Infinite ability to scale by adding more servers or disks.

28.2.2 Why Is This So Hard?

In short, the problem is one of *state*.

- All TCP/IP connections are dependent on state information.

The TCP connection involves a packet sequence number. This sequence number would need to be dynamically updated on all machines in the cluster to effect seamless TCP fail-over.

- CIFS/SMB (the Windows networking protocols) uses TCP connections.

This means that from a basic design perspective, fail-over is not seriously considered.

- All current SMB clusters are fail-over solutions — they rely on the clients to reconnect. They provide server fail-over, but clients can lose information due to a server failure.
- Servers keep state information about client connections.
 - CIFS/SMB involves a lot of state.
 - Every file open must be compared with other file opens to check share modes.

28.2.2.1 The Front-End Challenge

To make it possible for a cluster of file servers to appear as a single server that has one name and one IP address, the incoming TCP data streams from clients must be processed by the front end virtual server. This server must de-multiplex the incoming packets at the SMB protocol layer level and then feed the SMB packet to different servers in the cluster.

One could split all IPC\$ connections and RPC calls to one server to handle printing and user lookup requirements. RPC Printing handles are shared between different IPC4 sessions — it is hard to split this across clustered servers!

Conceptually speaking, all other servers would then provide only file services. This is a simpler problem to concentrate on.

28.2.2.2 De-multiplexing SMB Requests

De-multiplexing of SMB requests requires knowledge of SMB state information, all of which must be held by the front-end *virtual* server. This is a perplexing and complicated problem to solve.

Windows XP and later have changed semantics so state information (vuid, tid, fid) must match for a successful operation. This makes things simpler than before and is a positive step forward.

SMB requests are sent by vuid to their associated server. No code exists today to affect this solution. This problem is conceptually similar to the problem of correctly handling requests from multiple requests from Windows 2000 Terminal Server in Samba.

One possibility is to start by exposing the server pool to clients directly. This could eliminate the de-multiplexing step.

28.2.2.3 The Distributed File System Challenge

There exists many distributed file systems for UNIX and Linux.

Many could be adopted to backend our cluster, so long as awareness of SMB semantics is kept in mind (share modes, locking and oplock issues in particular). Common free distributed file systems include:

- NFS
- AFS
- OpenGFS
- Lustre

The server pool (cluster) can use any distributed file system backend if all SMB semantics are performed within this pool.

28.2.2.4 Restrictive Constraints on Distributed File Systems

Where a clustered server provides purely SMB services, oplock handling may be done within the server pool without imposing a need for this to be passed to the backend file system pool.

On the other hand, where the server pool also provides NFS or other file services, it will be essential that the implementation be oplock aware so it can interoperate with SMB services.

This is a significant challenge today. A failure to provide this will result in a significant loss of performance that will be sorely noted by users of Microsoft Windows clients.

Last, all state information must be shared across the server pool.

28.2.2.5 Server Pool Communications

Most backend file systems support POSIX file semantics. This makes it difficult to push SMB semantics back into the file system. POSIX locks have different properties and semantics from SMB locks.

All **smbd** processes in the server pool must of necessity communicate very quickly. For this, the current *tdb* file structure that Samba uses is not suitable for use across a network. Clustered **smbd**'s must use something else.

28.2.2.6 Server Pool Communications Demands

High speed inter-server communications in the server pool is a design prerequisite for a fully functional system. Possibilities for this include:

- Proprietary shared memory bus (example: Myrinet or SCI [Scalable Coherent Interface]). These are high cost items.
- Gigabit ethernet (now quite affordable).
- Raw ethernet framing (to bypass TCP and UDP overheads).

We have yet to identify metrics for performance demands to enable this to happen effectively.

28.2.2.7 Required Modifications to Samba

Samba needs to be significantly modified to work with a high-speed server inter-connect system to permit transparent fail-over clustering.

Particular functions inside Samba that will be affected include:

- The locking database, oplock notifications, and the share mode database.
- Failure semantics need to be defined. Samba behaves the same way as Windows. When oplock messages fail, a file open request is allowed, but this is potentially dangerous in a clustered environment. So how should inter-server pool failure semantics function and how should this be implemented?
- Should this be implemented using a point-to-point lock manager, or can this be done using multicast techniques?

28.2.3 A Simple Solution

Allowing fail-over servers to handle different functions within the exported file system removes the problem of requiring a distributed locking protocol.

If only one server is active in a pair, the need for high speed server interconnect is avoided. This allows the use of existing high availability solutions, instead of inventing a new one. This simpler solution comes at a price — the cost of which is the need to manage a more complex file name space. Since there is now not a single file system, administrators must remember where all services are located — a complexity not easily dealt with.

The *virtual server* is still needed to redirect requests to backend servers. Backend file space integrity is the responsibility of the administrator.

28.2.4 High Availability Server Products

Fail-over servers must communicate in order to handle resource fail-over. This is essential for high availability services. The use of a dedicated heartbeat is a common technique to introduce some intelligence into the fail-over process. This is often done over a dedicated link (LAN or serial).

Many fail-over solutions (like Red Hat Cluster Manager, as well as Microsoft Wolfpack) can use a shared SCSI or Fiber Channel disk storage array for fail-over communication. Information regarding Red Hat high availability solutions for Samba may be obtained from: www.redhat.com.¹

The Linux High Availability project is a resource worthy of consultation if your desire is to build a highly available Samba file server solution. Please consult the home page at www.linux-ha.org/.²

Front-end server complexity remains a challenge for high availability as it needs to deal gracefully with backend failures, while at the same time it needs to provide continuity of service to all network clients.

28.2.5 MS-DFS: The Poor Man's Cluster

MS-DFS links can be used to redirect clients to disparate backend servers. This pushes complexity back to the network client, something already included by Microsoft. MS-DFS creates the illusion of a simple, continuous file system name space, that even works at the file level.

Above all, at the cost of complexity of management, a distributed (pseudo-cluster) can be created using existing Samba functionality.

28.2.6 Conclusions

- Transparent SMB clustering is hard to do!
- Client fail-over is the best we can do today.
- Much more work is needed before a practical and manageable high availability transparent cluster solution will be possible.

¹<http://www.redhat.com/docs/manuals/enterprise/RHEL-AS-2.1-Manual/cluster-manager/s1-service-samba.html>

²<http://www.linux-ha.org/>

- MS-DFS can be used to create the illusion of a single transparent cluster.

Part IV

Migration and Updating

UPGRADING FROM SAMBA-2.X TO SAMBA-3.0.0

This chapter deals exclusively with the differences between Samba-3.0.0 and Samba-2.2.8a. It points out where configuration parameters have changed, and provides a simple guide for the move from 2.2.x to 3.0.0.

29.1 Quick Migration Guide

Samba-3.0.0 default behavior should be approximately the same as Samba-2.2.x. The default behavior when the new parameter *passdb backend* is not defined in the `smb.conf` file provides the same default behaviour as Samba-2.2.x with *encrypt passwords* = Yes, and will use the `smbpasswd` database.

So why say that *behavior should be approximately the same as Samba-2.2.x*? Because Samba-3.0.0 can negotiate new protocols, such as support for native Unicode, that may result in differing protocol code paths being taken. The new behavior under such circumstances is not exactly the same as the old one. The good news is that the domain and machine SIDs will be preserved across the upgrade.

If the Samba-2.2.x system was using an LDAP backend, and there is no time to update the LDAP database, then make sure that *passdb backend* = `ldapsam.compat` is specified in the `smb.conf` file. For the rest, behavior should remain more or less the same. At a later date, when there is time to implement a new Samba-3 compatible LDAP backend, it is possible to migrate the old LDAP database to the new one through use of the `pdbedit`. See Section 10.3.2.

29.2 New Features in Samba-3

The major new features are:

1. Active Directory support. This release is able to join an ADS realm as a member server and authenticate users using LDAP/kerberos.

2. Unicode support. Samba will now negotiate unicode on the wire and internally there is a much better infrastructure for multi-byte and unicode character sets.
3. New authentication system. The internal authentication system has been almost completely rewritten. Most of the changes are internal, but the new authoring system is also very configurable.
4. New filename mangling system. The filename mangling system has been completely rewritten. An internal database now stores mangling maps persistently.
5. New “*net*” command. A new “*net*” command has been added. It is somewhat similar to the “*net*” command in Windows. Eventually, we plan to replace a bunch of other utilities (such as `smbpasswd`) with subcommands in “*net*”.
6. Samba now negotiates NT-style status32 codes on the wire. This considerably improves error handling.
7. Better Windows 200x/XP printing support including publishing printer attributes in Active Directory.
8. New loadable RPC modules for `passdb` backends and character sets.
9. New default dual-daemon `winbindd` support for better performance.
10. Support for migrating from a Windows NT 4.0 domain to a Samba domain and maintaining user, group and domain SIDs.
11. Support for establishing trust relationships with Windows NT 4.0 Domain Controllers.
12. Initial support for a distributed Winbind architecture using an LDAP directory for storing SID to UID/GID mappings.
13. Major updates to the Samba documentation tree.
14. Full support for client and server SMB signing to ensure compatibility with default Windows 2003 security settings.

Plus lots of other improvements!

29.3 Configuration Parameter Changes

This section contains a brief listing of changes to `smb.conf` options in the 3.0.0 release. Please refer to the `smb.conf(5)` man page for complete descriptions of new or modified parameters.

29.3.1 Removed Parameters

(Ordered Alphabetically):

- `admin log`
- `alternate permissions`
- `character set`

- client codepage
- code page directory
- coding system
- domain admin group
- domain guest group
- force unknown acl user
- nt smb support
- post script
- printer driver
- printer driver file
- printer driver location
- status
- stip dot
- total print jobs
- use rhosts
- valid chars
- vfs options

29.3.2 New Parameters

(New parameters have been grouped by function):

Remote Management

- abort shutdown script
- shutdown script

User and Group Account Management:

- add group script
- add machine script
- add user to group script
- algorithmic rid base
- delete group script
- delete user from group script
- passdb backend
- set primary group script

Authentication:

- auth methods
- realm

Protocol Options:

- client lanman auth
- client NTLMv2 auth
- client schannel
- client signing
- client use spnego
- disable netbios
- ntlm auth
- paranoid server security
- server schannel
- server signing
- smb ports
- use spnego

File Service:

- get quota command
- hide special files
- hide unwriteable files
- hostname lookups
- kernel change notify
- mangle prefix
- map acl inherit
- msdfs proxy
- set quota command
- use sendfile
- vfs objects

Printing:

- max reported print jobs

Unicode and Character Sets:

- display charset

- dos charset
- unicode
- UNIX charset

SID to UID/GID Mappings:

- idmap backend
- idmap gid
- idmap uid
- winbind enable local accounts
- winbind trusted domains only
- template primary group
- enable rid algorithm

LDAP:

- ldap delete dn
- ldap group suffix
- ldap idmap suffix
- ldap machine suffix
- ldap passwd sync
- ldap user suffix

General Configuration:

- preload modules
- privatedir

29.3.3 Modified Parameters (Changes in Behavior):

- encrypt passwords (enabled by default)
- mangling method (set to hash2 by default)
- passwd chat
- passwd program
- password server
- restrict anonymous (integer value)
- security (new ads value)
- strict locking (enabled by default)
- winbind cache time (increased to 5 minutes)

- winbind uid (deprecated in favor of idmap uid)
- winbind gid (deprecated in favor of idmap gid)

29.4 New Functionality

29.4.1 Databases

This section contains brief descriptions of any new databases introduced in Samba-3. Please remember to backup your existing `lock directory`/*tdb before upgrading to Samba-3. Samba will upgrade databases as they are opened (if necessary), but downgrading from 3.0 to 2.2 is an unsupported path.

The new tdb files are described in Table 29.1.

Table 29.1. TDB File Descriptions

Name	Description	Backup?
account_policy	User policy settings	yes
gencache	Generic caching db	no
group_mapping	Mapping table from Windows groups/SID to UNIX groups	yes
idmap	new ID map table from SIDS to UNIX UIDs/GIDs	yes
namecache	Name resolution cache entries	no
netlogon_unigrp	Cache of universal group membership obtained when operating as a member of a Windows domain	no
printing/*tdb	Cached output from 'lpq command' created on a per print service basis	no
registry	Read-only Samba registry skeleton that provides support for exporting various db tables via the winreg RPCs	no

29.4.2 Changes in Behavior

The following issues are known changes in behavior between Samba-2.2 and Samba-3 that may affect certain installations of Samba.

1. When operating as a member of a Windows domain, Samba-2.2 would map any users authenticated by the remote DC to the *“guest account”* if a uid could not be obtained via the `getpwnam()` call. Samba-3 rejects the connection as `NT_STATUS_LOGON_FAILURE`. There is no current work around to re-establish the Samba-2.2 behavior.
2. When adding machines to a Samba-2.2 controlled domain, the *“add user script”* was used to create the UNIX identity of the Machine Trust Account. Samba-3 introduces a new *“add machine script”* that must be specified for this purpose. Samba-3 will not fall back to using the *“add user script”* in the absence of an *“add machine script”*.

29.4.3 Passdb Backends and Authentication

There have been a few new changes that Samba administrators should be aware of when moving to Samba-3.

1. Encrypted passwords have been enabled by default in order to interoperate better with out-of-the-box Windows client installations. This does mean that either (a) a Samba account must be created for each user, or (b) “*encrypt passwords = no*” must be explicitly defined in `smb.conf`.
2. Inclusion of new `security = ads` option for integration with an Active Directory domain using the native Windows Kerberos 5 and LDAP protocols.

Samba-3 also includes the possibility of setting up chains of authentication methods (*auth methods*) and account storage backends (*passdb backend*). Please refer to the `smb.conf` man page and Chapter 10, *Account Information Databases*, for details. While both parameters assume sane default values, it is likely that you will need to understand what the values actually mean in order to ensure Samba operates correctly.

Certain functions of the `smbpasswd` tool have been split between the new `smbpasswd` utility, the `net` tool and the new `pdbedit` utility. See the respective man pages for details.

29.4.4 LDAP

This section outlines the new features effecting Samba/LDAP integration.

29.4.4.1 New Schema

A new object class (`sambaSamAccount`) has been introduced to replace the old `sambaAccount`. This change aids us in the renaming of attributes to prevent clashes with attributes from other vendors. There is a conversion script (`examples/LDAP/convertSambaAccount`) to modify an LDIF file to the new schema.

Example:

```
$ ldapsearch .... -b "ou=people,dc=..." > old.ldif
$ convertSambaAccount <DOM SID> old.ldif new.ldif
```

The `<DOM SID>` can be obtained by running

```
$ net getlocalsid <DOMAINNAME>
```

on the Samba PDC as root.

The old `sambaAccount` schema may still be used by specifying the `ldapsam_compat` passdb backend. However, the `sambaAccount` and associated attributes have been moved to the historical section of the schema file and must be uncommented before use if needed. The

Samba-2.2 object class declaration for a sambaAccount has not changed in the Samba-3 samba.schema file.

Other new object classes and their uses include:

- sambaDomain — domain information used to allocate RIDs for users and groups as necessary. The attributes are added in “*ldap suffix*” directory entry automatically if an idmap UID/GID range has been set and the “*ldapsam*” passdb backend has been selected.
- sambaGroupMapping — an object representing the relationship between a posixGroup and a Windows group/SID. These entries are stored in the “*ldap group suffix*” and managed by the “*net groupmap*” command.
- sambaUNIXidPool — created in the “*ldap idmap suffix*” entry automatically and contains the next available “*idmap UID*” and “*idmap GID*”.
- sambaIdmapEntry — object storing a mapping between a SID and a UNIX UID/GID. These objects are created by the idmap_ldap module as needed.

29.4.4.2 New Suffix for Searching

The following new smb.conf parameters have been added to aid in directing certain LDAP queries when *passdb backend = ldapsam://...* has been specified.

- ldap suffix — used to search for user and computer accounts.
- ldap user suffix — used to store user accounts.
- ldap machine suffix — used to store Machine Trust Accounts.
- ldap group suffix — location of posixGroup/sambaGroupMapping entries.
- ldap idmap suffix — location of sambaIdmapEntry objects.

If an *ldap suffix* is defined, it will be appended to all of the remaining sub-suffix parameters. In this case, the order of the suffix listings in smb.conf is important. Always place the *ldap suffix* first in the list.

Due to a limitation in Samba’s smb.conf parsing, you should not surround the DNs with quotation marks.

29.4.4.3 IdMap LDAP Support

Samba-3 supports an ldap backend for the idmap subsystem. The following options inform Samba that the idmap table should be stored on the directory server ontrose in the “ou=idmap,dc=quenya,dc=org” partition.

```
...
idmap backend = ldap:ldap://ontrose/
ldap idmap suffix = ou=idmap,dc=quenya,dc=org
idmap uid = 40000-50000
idmap gid = 40000-50000
```

This configuration allows Winbind installations on multiple servers to share a UID/GID number space, thus avoiding the interoperability problems with NFS that were present in Samba-2.2.

MIGRATION FROM NT4 PDC TO SAMBA-3 PDC

This is a rough guide to assist those wishing to migrate from NT4 Domain Control to Samba-3-based Domain Control.

30.1 Planning and Getting Started

In the IT world there is often a saying that all problems are encountered because of poor planning. The corollary to this saying is that not all problems can be anticipated and planned for. Then again, good planning will anticipate most show-stopper-type situations.

Those wishing to migrate from MS Windows NT4 Domain Control to a Samba-3 Domain Control environment would do well to develop a detailed migration plan. So here are a few pointers to help migration get under way.

30.1.1 Objectives

The key objective for most organizations will be to make the migration from MS Windows NT4 to Samba-3 Domain Control as painless as possible. One of the challenges you may experience in your migration process may well be one of convincing management that the new environment should remain in place. Many who have introduced open source technologies have experienced pressure to return to a Microsoft-based platform solution at the first sign of trouble.

Before attempting a migration to a Samba-3 controlled network, make every possible effort to gain all-round commitment to the change. Know precisely *why* the change is important for the organization. Possible motivations to make a change include:

- Improve network manageability.
- Obtain better user level functionality.
- Reduce network operating costs.
- Reduce exposure caused by Microsoft withdrawal of NT4 support.

- Avoid MS License 6 implications.
- Reduce organization's dependency on Microsoft.

Make sure everyone knows that Samba-3 is not MS Windows NT4. Samba-3 offers an alternative solution that is both different from MS Windows NT4 and offers advantages compared with it. Gain recognition that Samba-3 lacks many of the features that Microsoft has promoted as core values in migration from MS Windows NT4 to MS Windows 2000 and beyond (with or without Active Directory services).

What are the features that Samba-3 cannot provide?

- Active Directory Server.
- Group Policy Objects (in Active Directory).
- Machine Policy Objects.
- Logon Scripts in Active Directory.
- Software Application and Access Controls in Active Directory.

The features that Samba-3 does provide and that may be of compelling interest to your site include:

- Lower cost of ownership.
- Global availability of support with no strings attached.
- Dynamic SMB Servers (can run more than one SMB/CIFS server per UNIX/Linux system).
- Creation of on-the-fly logon scripts.
- Creation of on-the-fly Policy Files.
- Greater stability, reliability, performance and availability.
- Manageability via an ssh connection.
- Flexible choices of back-end authentication technologies (tdbsam, ldapsam, mysqlsam).
- Ability to implement a full single-sign-on architecture.
- Ability to distribute authentication systems for absolute minimum wide area network bandwidth demand.

Before migrating a network from MS Windows NT4 to Samba-3, consider all necessary factors. Users should be educated about changes they may experience so the change will be a welcome one and not become an obstacle to the work they need to do. The following are factors that will help ensure a successful migration:

30.1.1.1 Domain Layout

Samba-3 can be configured as a Domain Controller, a back-up Domain Controller (probably best called a secondary controller), a Domain Member, or as a stand-alone Server. The

Windows network security domain context should be sized and scoped before implementation. Particular attention needs to be paid to the location of the primary Domain Controller (PDC) as well as backup controllers (BDCs). One way in which Samba-3 differs from Microsoft technology is that if one chooses to use an LDAP authentication backend, then the same database can be used by several different domains. In a complex organization, there can be a single LDAP database, which itself can be distributed (have a master server and multiple slave servers) that can simultaneously serve multiple domains.

>From a design perspective, the number of users per server as well as the number of servers per domain should be scaled taking into consideration server capacity and network bandwidth.

A physical network segment may house several domains. Each may span multiple network segments. Where domains span routed network segments, consider and test the performance implications of the design and layout of a network. A centrally located Domain Controller that is designed to serve multiple routed network segments may result in severe performance problems. Check the response time (ping timing) between the remote segment and the PDC. If it's long (more than 100 ms), locate a backup controller (BDC) on the remote segment to serve as the local authentication and access control server.

30.1.1.2 Server Share and Directory Layout

There are cardinal rules to effective network design that cannot be broken with impunity. The most important rule: Simplicity is king in every well-controlled network. Every part of the infrastructure must be managed; the more complex it is, the greater will be the demand of keeping systems secure and functional.

Keep in mind the nature of how data must be shared. Physical disk space layout should be considered carefully. Some data must be backed up. The simpler the disk layout the easier it will be to keep track of backup needs. Identify what backup media will meet your needs; consider backup to tape, CD-ROM or (DVD-ROM), or other offline storage medium. Plan and implement for minimum maintenance. Leave nothing to chance in your design; above all, do not leave backups to chance: Backup, test, and validate every backup, create a disaster recovery plan and prove that it works.

Users should be grouped according to data access control needs. File and directory access is best controlled via group permissions and the use of the “*sticky bit*” on group controlled directories may substantially avoid file access complaints from Samba share users.

Inexperienced network administrators often attempt elaborate techniques to set access controls on files, directories, shares, as well as in share definitions. Keep your design and implementation simple and document your design extensively. Have others audit your documentation. Do not create a complex mess that your successor will not understand. Remember, job security through complex design and implementation may cause loss of operations and downtime to users as the new administrator learns to untangle your knots. Keep access controls simple and effective and make sure that users will never be interrupted by obtuse complexity.

30.1.1.3 Logon Scripts

Logon scripts can help to ensure that all users gain the share and printer connections they need.

Logon scripts can be created on-the-fly so all commands executed are specific to the rights and privileges granted to the user. The preferred controls should be affected through group membership so group information can be used to create a custom logon script using the *root preexec* parameters to the *NETLOGON* share.

Some sites prefer to use a tool such as **kixstart** to establish a controlled user environment. In any case, you may wish to do a Google search for logon script process controls. In particular, you may wish to explore the use of the Microsoft KnowledgeBase article KB189105 that deals with how to add printers without user intervention via the logon script process.

30.1.1.4 Profile Migration/Creation

User and Group Profiles may be migrated using the tools described in the section titled Desktop Profile Management.

Profiles may also be managed using the Samba-3 tool **profiles**. This tool allows the MS Windows NT-style security identifiers (SIDs) that are stored inside the profile *NTuser.DAT* file to be changed to the SID of the Samba-3 domain.

30.1.1.5 User and Group Accounts

It is possible to migrate all account settings from an MS Windows NT4 domain to Samba-3. Before attempting to migrate user and group accounts, it is **STRONGLY** advised to create in Samba-3 the groups that are present on the MS Windows NT4 domain *AND* to map them to suitable UNIX/Linux groups. By following this simple advice, all user and group attributes should migrate painlessly.

30.1.2 Steps in Migration Process

The approximate migration process is described below.

- You have an NT4 PDC that has the users, groups, policies and profiles to be migrated.
- Samba-3 set up as a DC with netlogon share, profile share, and so on. Configure the *smb.conf* file to function as a BDC, i.e., *domain master = No*.

THE ACCOUNT MIGRATION PROCESS

1. Create a BDC account in the old NT4 domain for the Samba server using NT Server Manager.
 - (a) Samba must not be running.
2. `net rpc join -S NT4PDC -w DOMNAME -U Administrator%passwd`
3. `net rpc vampire -S NT4PDC -U administrator%passwd`

4. `pdbedit -L`

(a) Note — did the users migrate?

5. Now assign each of the UNIX groups to NT groups: (It may be useful to copy this text to a script called `initGroups.sh`)

```
#!/bin/bash
#### Keep this as a shell script for future re-use

# First assign well known domain global groups
net groupmap modify ntgroup="Domain Admins" unixgroup=root rid=512
net groupmap modify ntgroup="Domain Users" unixgroup=users rid=513
net groupmap modify ntgroup="Domain Guests" unixgroup=nobody rid=514

# Now for our added domain global groups
net groupmap add ntgroup="Designers" unixgroup=designers type=d rid=3200
net groupmap add ntgroup="Engineers" unixgroup=engineers type=d rid=3210
net groupmap add ntgroup="QA Team" unixgroup=qateam type=d rid=3220
```

6. `net groupmap list`

(a) Check that all groups are recognized.

Migrate all the profiles, then migrate all policy files.

30.2 Migration Options

Sites that wish to migrate from MS Windows NT4 Domain Control to a Samba-based solution generally fit into three basic categories. Table 30.1 shows the possibilities.

Table 30.1. The Three Major Site Types

Number of Users	Description
< 50	Want simple conversion with no pain.
50 - 250	Want new features, can manage some in-house complexity.
> 250	Solution/Implementation must scale well, complex needs. Cross-departmental decision process. Local expertise in most areas.

30.2.1 Planning for Success

There are three basic choices for sites that intend to migrate from MS Windows NT4 to Samba-3:

- Simple conversion (total replacement).
- Upgraded conversion (could be one of integration).
- Complete redesign (completely new solution).

Minimize down-stream problems by:

- Taking sufficient time.
- Avoiding Panic.
- Testing all assumptions.
- Testing the full roll-out program, including workstation deployment.

Table 30.2 lists the conversion choices given the type of migration being contemplated.

Table 30.2. Nature of the Conversion Choices

Simple	Upgraded	Redesign
Make use of minimal OS specific features.	Translate NT4 features to new host OS features.	Decide:
Move all accounts from NT4 into Samba-3	Copy and improve	Authentication regime (database location and access)
Make least number of operational changes	Make progressive improvements	Desktop management methods
Take least amount of time to migrate	Minimize user impact	Better control of Desktops/Users
Live versus isolated conversion	Maximize functionality	Identify Needs for: <i>Manageability, Scalability, Security, Availability</i>
Integrate Samba-3 then migrate while users are active, then change of control (swap out)	Take advantage of lower maintenance opportunity	

30.2.2 Samba-3 Implementation Choices

Authentication Database/Backend — Samba-3 can use an external authentication backend:

- Winbind (external Samba or NT4/200x server).
- External server could use Active Directory or NT4 Domain.
- Can use pam_mkhomedir.so to auto-create home dirs.
- Samba-3 can use a local authentication backend: *smbpasswd, tdbsam, ldap-sam, mysqlsam*

Access Control Points — Samba permits Access Control Points to be set:

- On the share itself — using Share ACLs.
- On the file system — using UNIX permissions on files and directories.

Note: Can enable Posix ACLs in file system also.

- Through Samba share parameters — not recommended except as last resort.

Policies (migrate or create new ones) — Exercise great caution when affecting registry changes, use the right tool and be aware that changes made through NT4-style `NTConfig.POL` files can leave permanent changes.

- Using Group Policy Editor (NT4).
- Watch out for Tattoo effect.

User and Group Profiles — Platform-specific so use platform tool to change from a Local to a Roaming profile. Can use new profiles tool to change SIDs (`NTUser.DAT`).

Logon Scripts — Know how they work.

User and Group Mapping to UNIX/Linux — User and Group mapping code is new. Many problems have been experienced as network administrators who are familiar with Samba-2.2.x migrate to Samba-3. Carefully study the chapters that document the new password backend behavior and the new group mapping functionality.

- The *username map* facility may be needed.
- Use **net groupmap** to connect NT4 groups to UNIX groups.
- Use **pdbedit** to set/change user configuration.

When migrating to LDAP backend, it may be easier to dump the initial LDAP database to LDIF, edit, then reload into LDAP.

OS Specific Scripts/Programs may be Needed — Every operating system has its peculiarities. These are the result of engineering decisions that were based on the experience of the designer, and may have side-effects that were not anticipated. Limitations that may bite the Windows network administrator include:

- Add/Delete Users: Note OS limits on size of name (Linux 8 chars) NT4 up to 254 chars.
- Add/Delete Machines: Applied only to Domain Members (Note: machine names may be limited to 16 characters).
- Use **net groupmap** to connect NT4 groups to UNIX groups.
- Add/Delete Groups: Note OS limits on size and nature. Linux limit is 16 char, no spaces and no upper case chars (**groupadd**).

Migration Tools — Domain Control (NT4 Style) Profiles, Policies, Access Controls, Security

- Samba: **net, rpcclient, smbpasswd, pdbedit, profiles.**
- Windows: **NT4 Domain User Manager, Server Manager (NEXUS)**

SWAT — THE SAMBA WEB ADMINISTRATION TOOL

There are many and varied opinions regarding the usefulness of SWAT. No matter how hard one tries to produce the perfect configuration tool, it remains an object of personal taste. SWAT is a tool that will allow Web-based configuration of Samba. It has a wizard that may help to get Samba configured quickly, it has context-sensitive help on each `smb.conf` parameter, it provides for monitoring of current state of connection information, and it allows network-wide MS Windows network password management.

31.1 Features and Benefits

SWAT is a facility that is part of the Samba suite. The main executable is called `swat` and is invoked by the inter-networking super daemon. See Section 31.2.2 for details.

SWAT uses integral samba components to locate parameters supported by the particular version of Samba. Unlike tools and utilities that are external to Samba, SWAT is always up to date as known Samba parameters change. SWAT provides context-sensitive help for each configuration parameter, directly from `man` page entries.

There are network administrators who believe that it is a good idea to write systems documentation inside configuration files, and for them SWAT will always be a nasty tool. SWAT does not store the configuration file in any intermediate form, rather, it stores only the parameter settings, so when SWAT writes the `smb.conf` file to disk, it will write only those parameters that are at other than the default settings. The result is that all comments, as well as parameters that are no longer supported, will be lost from the `smb.conf` file. Additionally, the parameters will be written back in internal ordering.

NOTE



Before using SWAT, please be warned — SWAT will completely replace your `smb.conf` with a fully-optimized file that has been stripped of all comments you might have placed there and only non-default settings will be written to the file.

31.2 Guidelines and Technical Tips

This section aims to unlock the dark secrets behind how SWAT may be made to work, may be made more secure, and how to solve Internationalization support problems.

31.2.1 Validate SWAT Installation

The very first step that should be taken before attempting to configure a host system for SWAT operation is to check that it is installed. This may seem a trivial point to some, however several Linux distributions do not install SWAT by default, even though they do ship an installable binary support package containing SWAT on the distribution media.

When you have confirmed that SWAT is installed it is necessary to validate that the installation includes the binary `swat` file as well as all the supporting text and Web files. A number of operating system distributions in the past have failed to include the necessary support files, even though the `swat` binary executable file was installed.

Finally, when you are sure that SWAT has been fully installed, please check the SWAT has been enabled in the control file for the internetworking super-daemon (`inetd` or `xinetd`) that is used on your operating system platform.

31.2.1.1 Locating the swat File

To validate that SWAT is installed, first locate the `swat` binary file on the system. It may be found under the following directories:

```
/usr/local/samba/bin — the default Samba location.  
/usr/sbin — the default location on most Linux systems.  
/opt/samba/bin
```

The actual location is much dependant on the choice of the operating system vendor, or as determined by the administrator who compiled and installed Samba.

There are a number methods that may be used to locate the `swat` binary file. The following methods may be helpful:

If `swat` is in your current operating system search path it will be easy to find it. You can ask what are the command-line options for `swat` as shown here:

```
frodo:~ # swat -?
```



```
Usage: swat [OPTION...]
  -a, --disable-authentication    Disable authentication (demo mode)

Help options:
  -?, --help                      Show this help message
  --usage                          Display brief usage message

Common samba options:
  -d, --debuglevel=DEBUGLEVEL    Set debug level
  -s, --configfile=CONFIGFILE    Use alternative configuration file
  -l, --log-basename=LOGFILEBASE Basename for log/debug files
  -V, --version                   Print version
```

31.2.1.2 Locating the SWAT Support Files

Now that you have found that **swat** is in the search path, it is easy to identify where the file is located. Here is another simple way this may be done:

```
frodo:~ # whereis swat
swat: /usr/sbin/swat /usr/share/man/man8/swat.8.gz
```

If the above measures fail to locate the **swat** binary, another approach is needed. The following may be used:

```
frodo:/ # find / -name swat -print
/etc/xinetd.d/swat
/usr/sbin/swat
/usr/share/samba/swat
frodo:/ #
```

This list shows that there is a control file for **xinetd**, the internet network super-daemon that is installed on this server. The location of the SWAT binary file is `/usr/sbin/swat`, and the support files for it are located under the directory `/usr/share/samba/swat`.

We must now check where **swat** expects to find its support files. This can be done as follows:

```
frodo:/ # strings /usr/sbin/swat | grep "/swat"
/swat/
...
/usr/share/samba/swat
frodo:/ #
```

The `/usr/share/samba/swat/` entry shown in this listing is the location of the support files. You should verify that the support files exist under this directory. A sample list is as

shown:

```
jht@frodo: /> find /usr/share/samba/swat -print
/usr/share/samba/swat
/usr/share/samba/swat/help
/usr/share/samba/swat/lang
/usr/share/samba/swat/lang/ja
/usr/share/samba/swat/lang/ja/help
/usr/share/samba/swat/lang/ja/help/welcome.html
/usr/share/samba/swat/lang/ja/images
/usr/share/samba/swat/lang/ja/images/home.gif
...
/usr/share/samba/swat/lang/ja/include
/usr/share/samba/swat/lang/ja/include/header.nocss.html
...
/usr/share/samba/swat/lang/tr
/usr/share/samba/swat/lang/tr/help
/usr/share/samba/swat/lang/tr/help/welcome.html
/usr/share/samba/swat/lang/tr/images
/usr/share/samba/swat/lang/tr/images/home.gif
...
/usr/share/samba/swat/lang/tr/include
/usr/share/samba/swat/lang/tr/include/header.html
/usr/share/samba/swat/using_samba
...
/usr/share/samba/swat/images
/usr/share/samba/swat/images/home.gif
...
/usr/share/samba/swat/include
/usr/share/samba/swat/include/footer.html
/usr/share/samba/swat/include/header.html
jht@frodo: />
```

If the files needed are not available it will be necessary to obtain and install them before SWAT can be used.

31.2.2 Enabling SWAT for Use

SWAT should be installed to run via the network super-daemon. Depending on which system your UNIX/Linux system has, you will have either an **inetd**- or **xinetd**-based system.

The nature and location of the network super-daemon varies with the operating system implementation. The control file (or files) can be located in the file `/etc/inetd.conf` or in the directory `/etc/[x]inet[d].d` or similar.

The control entry for the older style file might be:

```
# swat is the Samba Web Administration Tool
swat stream tcp nowait.400 root /usr/sbin/swat swat
```

A control file for the newer style xinetd could be:

```
# default: off
# description: SWAT is the Samba Web Admin Tool. Use swat \
#             to configure your Samba server. To use SWAT, \
#             connect to port 901 with your favorite web browser.
service swat
{
    port      = 901
    socket_type = stream
    wait      = no
    only_from = localhost
    user      = root
    server    = /usr/sbin/swat
    log_on_failure += USERID
    disable   = yes
}
```

Both of the above examples assume that the **swat** binary has been located in the **/usr/sbin** directory. In addition to the above, SWAT will use a directory access point from which it will load its Help files as well as other control information. The default location for this on most Linux systems is in the directory **/usr/share/samba/swat**. The default location using Samba defaults will be **/usr/local/samba/swat**.

Access to SWAT will prompt for a logon. If you log onto SWAT as any non-root user, the only permission allowed is to view certain aspects of configuration as well as access to the password change facility. The buttons that will be exposed to the non-root user are: **HOME**, **STATUS**, **VIEW**, **PASSWORD**. The only page that allows change capability in this case is **PASSWORD**.

As long as you log onto SWAT as the user *root*, you should obtain full change and commit ability. The buttons that will be exposed include: **HOME**, **GLOBALS**, **SHARES**, **PRINTERS**, **WIZARD**, **STATUS**, **VIEW**, **PASSWORD**.

31.2.3 Securing SWAT through SSL

Many people have asked about how to setup SWAT with SSL to allow for secure remote administration of Samba. Here is a method that works, courtesy of Markus Krieger.

Modifications to the SWAT setup are as follows:

1. Install OpenSSL.
2. Generate certificate and private key.

```
root# /usr/bin/openssl req -new -x509 -days 365 -nodes -config \
    /usr/share/doc/packages/stunnel/stunnel.cnf \
    -out /etc/stunnel/stunnel.pem -keyout /etc/stunnel/stunnel.pem
```

3. Remove `swat`-entry from `[x]inetd`.
4. Start `stunnel`.

```
root# stunnel -p /etc/stunnel/stunnel.pem -d 901 \
    -l /usr/local/samba/bin/swat swat
```

Afterward, simply connect to `swat` by using the URL `https://myhost:901`, accept the certificate and the SSL connection is up.

31.2.4 Enabling SWAT Internationalization Support

SWAT can be configured to display its messages to match the settings of the language configurations of your Web browser. It will be passed to SWAT in the `Accept-Language` header of the HTTP request.

To enable this feature:

- Install the proper `msg` files from the Samba `source/po` directory into `$LIBDIR`.
- Set the correct locale value for *display charset*.
- Set your browser's language setting.

The name of `msg` file is same as the language ID sent by the browser. For example `en` means "English", `ja` means "Japanese", `fr` means "French".

If you do not like some of messages, or there are no `msg` files for your locale, you can create them simply by copying the `en.msg` files to the directory for "*your language ID.msg*" and filling in proper strings to each "*msgstr*". For example, in `it.msg`, the `msg` file for the Italian locale, just set:

```
msgid "Set Default"
msgstr "Imposta Default"
```

and so on. If you find a mistake or create a new `msg` file, please email it to us so we will include this in the next release of Samba.

Note that if you enable this feature and the *display charset* is not matched to your browser's setting, the SWAT display may be corrupted. In a future version of Samba, SWAT will always display messages with UTF-8 encoding. You will then not need to set this `smb.conf` file parameter.

31.3 Overview and Quick Tour

SWAT is a tools that many be used to configure Samba, or just to obtain useful links to important reference materials such as the contents of this book, as well as other documents that have been found useful for solving Windows networking problems.

31.3.1 The SWAT Home Page

The SWAT title page provides access to the latest Samba documentation. The manual page for each Samba component is accessible from this page, as are the Samba HOWTO-Collection (this document) as well as the O'Reilly book “*Using Samba*.”

Administrators who wish to validate their Samba configuration may obtain useful information from the man pages for the diagnostic utilities. These are available from the SWAT home page also. One diagnostic tool that is not mentioned on this page, but that is particularly useful is **ethereal**.¹

WARNING



SWAT can be configured to run in *demo* mode. This is not recommended as it runs SWAT without authentication and with full administrative ability. Allows changes to `smb.conf` as well as general operation with root privileges. The option that creates this ability is the `-a` flag to `swat`. *Do not use this in a production environment.*

31.3.2 Global Settings

The **GLOBALS** button will expose a page that allows configuration of the global parameters in `smb.conf`. There are two levels of exposure of the parameters:

- **Basic** — exposes common configuration options.
- **Advanced** — exposes configuration options needed in more complex environments.

To switch to other than **Basic** editing ability, click on **Advanced**. You may also do this by clicking on the radio button, then click on the **Commit Changes** button.

After making any changes to configuration parameters, make sure that you click on the **Commit Changes** button before moving to another area, otherwise your changes will be lost.

¹<http://www.ethereal.com/>

NOTE

SWAT has context-sensitive help. To find out what each parameter is for, simply click on the **Help** link to the left of the configuration parameter.

31.3.3 Share Settings

To effect a currently configured share, simply click on the pull down button between the **Choose Share** and the **Delete Share** buttons, select the share you wish to operate on, then to edit the settings click on the **Choose Share** button. To delete the share, simply press the **Delete Share** button.

To create a new share, next to the button labeled **Create Share** enter into the text field the name of the share to be created, then click on the **Create Share** button.

31.3.4 Printers Settings

To affect a currently configured printer, simply click on the pull down button between the **Choose Printer** and the **Delete Printer** buttons, select the printer you wish to operate on, then to edit the settings click on the **Choose Printer** button. To delete the share, simply press the **Delete Printer** button.

To create a new printer, next to the button labeled **Create Printer** enter into the text field the name of the share to be created, then click on the **Create Printer** button.

31.3.5 The SWAT Wizard

The purpose of the SWAT Wizard is to help the Microsoft-knowledgeable network administrator to configure Samba with a minimum of effort.

The Wizard page provides a tool for rewriting the `smb.conf` file in fully optimized format. This will also happen if you press the **Commit** button. The two differ since the **Rewrite** button ignores any changes that may have been made, while the **Commit** button causes all changes to be affected.

The **Edit** button permits the editing (setting) of the minimal set of options that may be necessary to create a working Samba server.

Finally, there are a limited set of options that will determine what type of server Samba will be configured for, whether it will be a WINS server, participate as a WINS client, or operate with no WINS support. By clicking one button, you can elect to expose (or not) user home directories.

31.3.6 The Status Page

The status page serves a limited purpose. First, it allows control of the Samba daemons. The key daemons that create the Samba server environment are: `smbd`, `nmbd`, `winbindd`.

The daemons may be controlled individually or as a total group. Additionally, you may set an automatic screen refresh timing. As MS Windows clients interact with Samba, new `smbd` processes will be continually spawned. The auto-refresh facility will allow you to track the changing conditions with minimal effort.

Lastly, the Status page may be used to terminate specific `smbd` client connections in order to free files that may be locked.

31.3.7 The View Page

This page allows the administrator to view the optimized `smb.conf` file and, if you are particularly masochistic, will permit you also to see all possible global configuration parameters and their settings.

31.3.8 The Password Change Page

The Password Change page is a popular tool that allows the creation, deletion, deactivation, and reactivation of MS Windows networking users on the local machine. Alternately, you can use this tool to change a local password for a user account.

When logged in as a non-root account, the user will have to provide the old password as well as the new password (twice). When logged in as `root`, only the new password is required.

One popular use for this tool is to change user passwords across a range of remote MS Windows servers.

31.4 SWAT View Page Displays Incorrectly

When `display charset` and `dos charset` parameters are different, the view page will not display correctly. Currently the `display charset` parameter must use the same encoding as that in which the `msg` file has been encoded. In Japanese this means that `display charset` must be set to `CP932`.

Setting `unix charset = EUCJP-MS` will cause this problem to occur.

Part V

Troubleshooting

THE SAMBA CHECKLIST

32.1 Introduction

This file contains a list of tests you can perform to validate your Samba server. It also tells you what the likely cause of the problem is if it fails any one of these steps. If it passes all these tests, then it is probably working fine.

You should do all the tests, in the order shown. We have tried to carefully choose them so later tests only use capabilities verified in the earlier tests. However, do not stop at the first error as there have been some instances when continuing with the tests has helped to solve a problem.

If you send one of the Samba mailing lists an email saying, “*it does not work*” and you have not followed this test procedure, you should not be surprised if your email is ignored.

32.2 Assumptions

In all of the tests, it is assumed you have a Samba server called BIGSERVER and a PC called ACLIENT both in workgroup TESTGROUP.

The procedure is similar for other types of clients.

It is also assumed you know the name of an available share in your `smb.conf`. I will assume this share is called `tmp`. You can add a `tmp` share like this by adding the lines shown in Example 32.1.

Example 32.1. `smb.conf` with `[tmp]` share

```
[tmp]
comment = temporary files
path = /tmp
read only = yes
```

NOTE



These tests assume version 3.0.0 or later of the Samba suite. Some commands shown did not exist in earlier versions.

Please pay attention to the error messages you receive. If any error message reports that your server is being unfriendly, you should first check that your IP name resolution is correctly set up. Make sure your `/etc/resolv.conf` file points to name servers that really do exist.

Also, if you do not have DNS server access for name resolution, please check that the settings for your `smb.conf` file results in `dns proxy = no`. The best way to check this is with `testparm smb.conf`.

It is helpful to monitor the log files during testing by using the `tail -F log_file_name` in a separate terminal console (use ctrl-alt-F1 through F6 or multiple terminals in X). Relevant log files can be found (for default installations) in `/usr/local/samba/var`. Also, connection logs from machines can be found here or possibly in `/var/log/samba`, depending on how or if you specified logging in your `smb.conf` file.

If you make changes to your `smb.conf` file while going through these test, remember to restart `smbd` and `nmbd`.

32.3 The Tests

DIAGNOSING YOUR SAMBA SERVER

1. In the directory in which you store your `smb.conf` file, run the command `testparm smb.conf`. If it reports any errors, then your `smb.conf` configuration file is faulty.

NOTE



Your `smb.conf` file may be located in: `/etc/samba` or in `/usr/local/samba/lib`.

2. Run the command `ping BIGSERVER` from the PC and `ping ACLIENT` from the UNIX box. If you do not get a valid response, then your TCP/IP software is not correctly installed. You will need to start a “*dos prompt*” window on the PC to run ping. If you get a message saying “*host not found*” or similar, then your DNS software or `/etc/hosts` file is not correctly setup. It is possible to run Samba without DNS entries for the server and client, but it is assumed you do have correct entries for the remainder of these tests. Another reason why ping might fail is if your host is running firewall software. You will need to relax the rules to let in the workstation in

question, perhaps by allowing access from another subnet (on Linux this is done via the appropriate firewall maintenance commands **ipchains** or **iptables**).

NOTE



Modern Linux distributions install ipchains/iptables by default. This is a common problem that is often overlooked.

If you wish to check what firewall rules may be present in a system under test, simply run **iptables -L -v** or if *ipchains*-based firewall rules are in use, **ipchains -L -v**. Here is a sample listing from a system that has an external ethernet interface (eth1) on which Samba is not active, and an internal (private network) interface (eth0) on which Samba is active:

```
frodo:~ # iptables -L -v
Chain INPUT (policy DROP 98496 packets, 12M bytes)
  pkts bytes target    prot opt in     out     source    destination
  187K  109M ACCEPT    all  --  lo      any     anywhere  anywhere
  892K  125M ACCEPT    all  --  eth0    any     anywhere  anywhere
  1399K 1380M ACCEPT    all  --  eth1    any     anywhere  anywhere \
      state RELATED,ESTABLISHED

Chain FORWARD (policy DROP 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source    destination
  978K 1177M ACCEPT    all  --  eth1    eth0    anywhere  anywhere \
      state RELATED,ESTABLISHED
  658K   40M ACCEPT    all  --  eth0    eth1    anywhere  anywhere
    0     0 LOG        all  --  any     any     anywhere  anywhere \
      LOG level warning

Chain OUTPUT (policy ACCEPT 2875K packets, 1508M bytes)
  pkts bytes target    prot opt in     out     source    destination

Chain reject_func (0 references)
  pkts bytes target    prot opt in     out     source    destinat
```

- Run the command: **smbclient -L BIGSERVER** on the UNIX box. You should get back a list of available shares. If you get an error message containing the string “*Bad password*”, then you probably have either an incorrect *hosts allow*, *hosts deny* or *valid users* line in your *smb.conf*, or your guest account is not valid. Check what your guest account is using *testparm* and temporarily remove any *hosts allow*, *hosts deny*, *valid users* or *invalid users* lines. If you get a message “*connection refused*” response, then the **smbd** server may not be running. If you installed it in *inetd.conf*, then you probably edited that file incorrectly. If you installed it as a

daemon, then check that it is running, and check that the netbios-ssn port is in a LISTEN state using **netstat -a**.

NOTE



Some UNIX/Linux systems use **xinetd** in place of **inetd**. Check your system documentation for the location of the control files for your particular system implementation of the network super daemon.

If you get a message saying “*session request failed*”, the server refused the connection. If it says “*Your server software is being unfriendly*”, then it’s probably because you have invalid command line parameters to `smbd`, or a similar fatal problem with the initial startup of `smbd`. Also check your config file (`smb.conf`) for syntax errors with `testparm` and that the various directories where Samba keeps its log and lock files exist. There are a number of reasons for which `smbd` may refuse or decline a session request. The most common of these involve one or more of the `smb.conf` file entries as shown in Example 32.2.

Example 32.2. Configuration for only allowing connections from a certain subnet

```
[globals]
...
hosts deny = ALL
hosts allow = xxx.xxx.xxx.xxx/yy
interfaces = eth0
bind interfaces only = Yes
...
```

In the above, no allowance has been made for any session requests that will automatically translate to the loopback adapter address 127.0.0.1. To solve this problem, change these lines as shown in Example 32.3.

Example 32.3. Configuration for allowing connections from a certain subnet and localhost

```
[globals]
...
hosts deny = ALL
hosts allow = xxx.xxx.xxx.xxx/yy 127.
interfaces = eth0 lo
...
```

Another common cause of these two errors is having something already running on port 139, such as Samba (smbd is running from inetd already) or something like Digital's Pathworks. Check your `inetd.conf` file before trying to start smbd as a daemon — it can avoid a lot of frustration! And yet another possible cause for failure of this test is when the subnet mask and/or broadcast address settings are incorrect. Please check that the network interface IP Address/Broadcast Address/Subnet Mask settings are correct and that Samba has correctly noted these in the `log.nmbd` file.

4. Run the command: **nmblookup -B BIGSERVER __SAMBA__**. You should get back the IP address of your Samba server. If you do not, then nmbd is incorrectly installed. Check your `inetd.conf` if you run it from there, or that the daemon is running and listening to udp port 137. One common problem is that many inetd implementations can't take many parameters on the command line. If this is the case, then create a one-line script that contains the right parameters and run that from inetd.
5. Run the command: **nmblookup -B ACLIENT '*'** You should get the PC's IP address back. If you do not then the client software on the PC isn't installed correctly, or isn't started, or you got the name of the PC wrong. If ACLIENT does not resolve via DNS then use the IP address of the client in the above test.
6. Run the command: **nmblookup -d 2 '*'** This time we are trying the same as the previous test but are trying it via a broadcast to the default broadcast address. A number of NetBIOS/TCP/IP hosts on the network should respond, although Samba may not catch all of the responses in the short time it listens. You should see the "*got a positive name query response*" messages from several hosts. If this does not give a similar result to the previous test, then nmblookup isn't correctly getting your broadcast address through its automatic mechanism. In this case you should experiment with the *interfaces* option in `smb.conf` to manually configure your IP address, broadcast and netmask. If your PC and server aren't on the same subnet, then you will need to use the -B option to set the broadcast address to that of the PCs subnet. This test will probably fail if your subnet mask and broadcast address are not correct. (Refer to TEST 3 notes above).
7. Run the command: **smbclient //BIGSERVER/TMP**. You should then be prompted for a password. You should use the password of the account with which you are logged into the UNIX box. If you want to test with another account, then add the -U `accountname` option to the end of the command line. For example, **smbclient //bigserver/tmp -Ujohndoe**.

NOTE

It is possible to specify the password along with the username as follows: **smbclient //bigserver/tmp -Ujohndoe%secret**.

Once you enter the password, you should get the `smb>` prompt. If you do not, then look at the error message. If it says "*invalid network name*", then the service `tmp` is

not correctly setup in your `smb.conf`. If it says “*bad password*”, then the likely causes are:

- (a) You have shadow passwords (or some other password system) but didn’t compile in support for them in `smbd`.
- (b) Your *valid users* configuration is incorrect.
- (c) You have a mixed case password and you haven’t enabled the *password level* option at a high enough level.
- (d) The *path* line in `smb.conf` is incorrect. Check it with `testparm`.
- (e) You enabled password encryption but didn’t map UNIX to Samba users. Run:
smbpasswd -a username

Once connected, you should be able to use the commands **dir**, **get**, **put** and so on. Type **help command** for instructions. You should especially check that the amount of free disk space shown is correct when you type **dir**.

8. On the PC, type the command **net view \\BIGSERVER**. You will need to do this from within a dos prompt window. You should get back a list of shares available on the server. If you get a message “*network name not found*” or similar error, then netbios name resolution is not working. This is usually caused by a problem in `nmbd`. To overcome it, you could do one of the following (you only need to choose one of them):
 - (a) Fixup the `nmbd` installation.
 - (b) Add the IP address of BIGSERVER to the **wins server** box in the advanced TCP/IP setup on the PC.
 - (c) Enable Windows name resolution via DNS in the advanced section of the TCP/IP setup.
 - (d) Add BIGSERVER to your `lmhosts` file on the PC.

If you get a message “*invalid network name*” or “*bad password error*”, then apply the same fixes as for the **smbclient -L** test above. In particular, make sure your **hosts allow** line is correct (see the man pages). Also, do not overlook that fact that when the workstation requests the connection to the Samba server, it will attempt to connect using the name with which you logged onto your Windows machine. You need to make sure that an account exists on your Samba server with that exact same name and password. If you get a message “*specified computer is not receiving requests*” or similar, it probably means that the host is not contactable via TCP services. Check to see if the host is running TCP wrappers, and if so add an entry in the `hosts.allow` file for your client (or subnet, and so on.)

9. Run the command **net use x: \\BIGSERVER\TMP**. You should be prompted for a password, then you should get a **command completed successfully** message. If not, then your PC software is incorrectly installed or your `smb.conf` is incorrect. Make sure your *hosts allow* and other config lines in `smb.conf` are correct. It’s also possible that the server can’t work out what user name to connect you as. To see if this is the problem, add the line *user* = username to the `[tmp]` section of `smb.conf` where *username* is the username corresponding to the password you typed. If

you find this fixes things, you may need the username mapping option. It might also be the case that your client only sends encrypted passwords and you have *encrypt passwords* = no in *smb.conf*. Change this to "yes" to fix this.

10. Run the command **nmblookup -M *testgroup*** where *testgroup* is the name of the workgroup that your Samba server and Windows PCs belong to. You should get back the IP address of the master browser for that workgroup. If you do not, then the election process has failed. Wait a minute to see if it is just being slow, then try again. If it still fails after that, then look at the browsing options you have set in *smb.conf*. Make sure you have *preferred master* = yes to ensure that an election is held at startup.
11. >From file manager, try to browse the server. Your Samba server should appear in the browse list of your local workgroup (or the one you specified in *smb.conf*). You should be able to double click on the name of the server and get a list of shares. If you get the error message "*invalid password*", you are probably running Windows NT and it is refusing to browse a server that has no encrypted password capability and is in User Level Security mode. In this case, either set *security* = server and *password server* = Windows_NT_Machine in your *smb.conf* file, or make sure *encrypt passwords* is set to "*yes*".

ANALYZING AND SOLVING SAMBA PROBLEMS

There are many sources of information available in the form of mailing lists, RFCs and documentation. The documentation that comes with the Samba distribution contains good explanations of general SMB topics such as browsing.

33.1 Diagnostics Tools

With SMB networking, it is often not immediately clear what the cause is of a certain problem. Samba itself provides rather useful information, but in some cases you might have to fall back to using a *sniffer*. A sniffer is a program that listens on your LAN, analyzes the data sent on it and displays it on the screen.

33.1.1 Debugging with Samba Itself

One of the best diagnostic tools for debugging problems is Samba itself. You can use the `-d` option for both `smbd` and `nmbd` to specify the *debug level* at which to run. See the man pages for `smbd`, `nmbd` and `smb.conf` for more information regarding debugging options. The debug level can range from 1 (the default) to 10 (100 for debugging passwords).

Another helpful method of debugging is to compile Samba using the `gcc -g` flag. This will include debug information in the binaries and allow you to attach `gdb` to the running `smbd/nmbd` process. To attach `gdb` to an `smbd` process for an NT workstation, first get the workstation to make the connection. Pressing `ctrl-alt-delete` and going down to the domain box is sufficient (at least, the first time you join the domain) to generate a `LsaEnumTrustedDomains`. Thereafter, the workstation maintains an open connection and there will be an `smbd` process running (assuming that you haven't set a really short `smbd` idle timeout). So, in between pressing `ctrl-alt-delete` and actually typing in your password, you can attach `gdb` and continue.

Some useful Samba commands worth investigating are:

```
$ testparm | more
```

```
$ smbclient -L //{netbios name of server}
```

33.1.2 Tcpcdump

Tcpcdump¹ was the first UNIX sniffer with SMB support. It is a command-line utility and now, its SMB support is somewhat lagging that of **ethereal** and **tethereal**.

33.1.3 Ethereal

Ethereal² is a graphical sniffer, available for both UNIX (Gtk) and Windows. Ethereal's SMB support is quite good.

For details on the use of **ethereal**, read the well-written Ethereal User Guide.

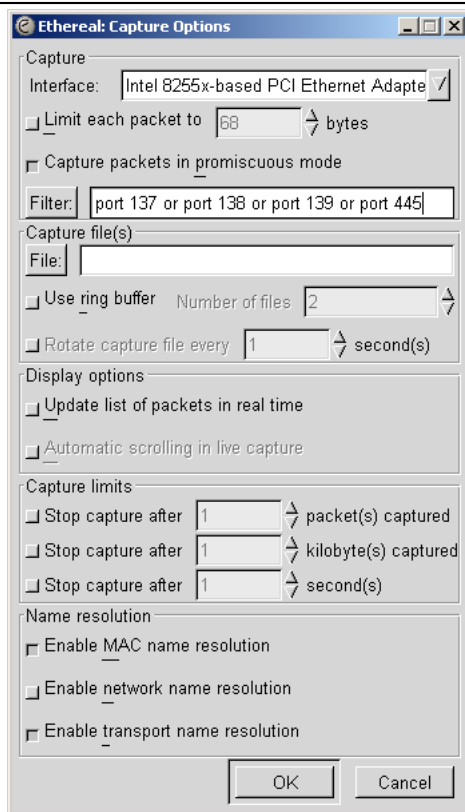


Figure 33.1. Starting a capture.

Listen for data on ports 137, 138, 139, and 445. For example, use the filter port 137, port 138, port 139, or port 445 as seen in Figure 33.1.

¹<http://www.tcpdump.org/>

²<http://www.ethereal.com/>

A console version of ethereal is available as well and is called **tethereal**.

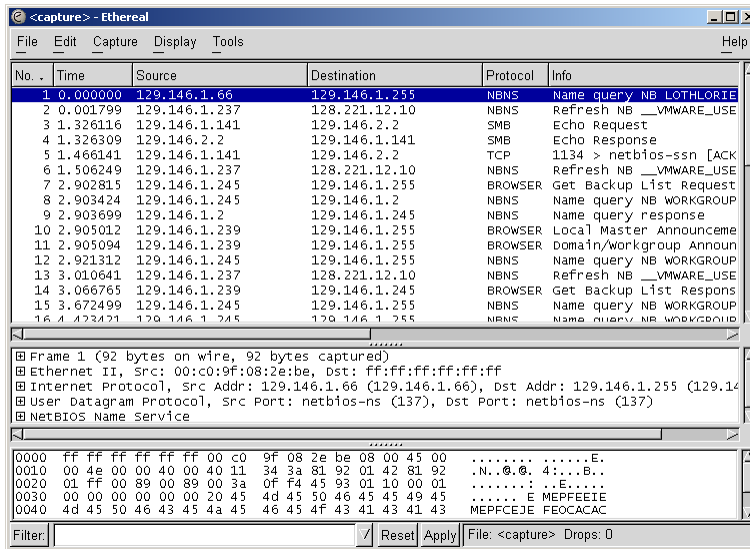


Figure 33.2. Main ethereal data window.

33.1.4 The Windows Network Monitor

For tracing things on Microsoft Windows NT, Network Monitor (aka Netmon) is available on Microsoft Developer Network CDs, the Windows NT Server install CD and the SMS CDs. The version of Netmon that ships with SMS allows for dumping packets between any two computers (i.e., placing the network interface in promiscuous mode). The version on the NT Server install CD will only allow monitoring of network traffic directed to the local NT box and broadcasts on the local subnet. Be aware that Ethereal can read and write Netmon formatted files.

33.1.4.1 Installing Network Monitor on an NT Workstation

Installing Netmon on an NT workstation requires a couple of steps. The following are instructions for installing Netmon V4.00.349, which comes with Microsoft Windows NT Server 4.0, on Microsoft Windows NT Workstation 4.0. The process should be similar for other versions of Windows NT version of Netmon. You will need both the Microsoft Windows NT Server 4.0 Install CD and the Workstation 4.0 Install CD.

Initially you will need to install Network Monitor Tools and Agent on the NT Server to do this:

- Go to **Start -> Settings -> Control Panel -> Network -> Services -> Add**.
- Select the **Network Monitor Tools and Agent** and click on **OK**.
- Click on **OK** on the Network Control Panel.

- Insert the Windows NT Server 4.0 install CD when prompted.

At this point, the Netmon files should exist in %SYSTEMROOT%\System32\netmon*.*. Two subdirectories exist as well, `parsers\` which contains the necessary DLLs for parsing the Netmon packet dump, and `captures\`.

To install the Netmon tools on an NT Workstation, you will first need to install the Network Monitor Agent from the Workstation install CD.

- Go to **Start -> Settings -> Control Panel -> Network -> Services -> Add**.
- Select the **Network Monitor Agent**, click on **OK**.
- Click on **OK** in the Network Control Panel.
- Insert the Windows NT Workstation 4.0 install CD when prompted.

Now copy the files from the NT Server in %SYSTEMROOT%\System32\netmon to %SYSTEMROOT%\System32\netmon on the Workstation and set permissions as you deem appropriate for your site. You will need administrative rights on the NT box to run Netmon.

33.1.4.2 Installing Network Monitor on Windows 9x/Me

To install Netmon on Windows 9x/Me, install the Network Monitor Agent from the Windows 9x/Me CD (`\admin\nettools\netmon`). There is a readme file located with the Netmon driver files on the CD if you need information on how to do this. Copy the files from a working Netmon installation.

33.2 Useful URLs

- See how Scott Merrill simulates a BDC behavior at <http://www.skippy.net/linux/smb-howto.html>.
- FTP site for older SMB specs: <ftp://ftp.microsoft.com/developr/drg/CIFS/>

33.3 Getting Mailing List Help

There are a number of Samba-related mailing lists. Go to <http://samba.org>, click on your nearest mirror and then click on **Support** and next click on **Samba-related mailing lists**.

For questions relating to Samba TNG, go to <http://www.samba-tng.org/>. It has been requested that you do not post questions about Samba-TNG to the main-stream Samba lists.

If you do post a message to one of the lists, please observe the following guidelines :

- Always remember that the developers are volunteers, they are not paid and they never guarantee to produce a particular feature at a particular time. Any timelines are “*best guess*” and nothing more.

- Always mention what version of Samba you are using and what operating system it's running under. You should list the relevant sections of your `smb.conf` file, at least the options in `[global]` that affect PDC support.
- In addition to the version, if you obtained Samba via CVS, mention the date when you last checked it out.
- Try and make your questions clear and brief. Lots of long, convoluted questions get deleted before they are completely read! Do not post HTML encoded messages. Most people on mailing lists simply delete them.
- If you run one of those nifty “*I'm on holidays*” things when you are away, make sure its configured to not answer mailing list traffic. Auto-responses to mailing lists really irritate the thousands of people who end up having to deal with such bad netiquet behavior.
- Don't cross post. Work out which is the best list to post to and see what happens. Do not post to both `samba-ntdom` and `samba-technical`. Many people active on the lists subscribe to more than one list and get annoyed to see the same message two or more times. Often someone will see a message and thinking it would be better dealt with on another list, will forward it on for you.
- You might include *partial* log files written at a debug level set to as much as 20. Please do not send the entire log but just enough to give the context of the error messages.
- If you have a complete Netmon trace (from the opening of the pipe to the error), you can send the `*.CAP` file as well.
- Please think carefully before attaching a document to an email. Consider pasting the relevant parts into the body of the message. The Samba mailing lists go to a huge number of people. Do they all need a copy of your `smb.conf` in their attach directory?

33.4 How to Get Off the Mailing Lists

To have your name removed from a Samba mailing list, go to the same place where you went to subscribe to it. Go to <http://lists.samba.org>, click on your nearest mirror, click on **Support** and then click on **Samba related mailing lists**.

Please do not post messages to the list asking to be removed. You will only be referred to the above address (unless that process failed in some way).

REPORTING BUGS

34.1 Introduction

Please report bugs using Samba's Bugzilla¹ facilities and take the time to read this file before you submit a bug report. Also, check to see if it has changed between releases, as we may be changing the bug reporting mechanism at some point.

Please do as much as you can yourself to help track down the bug. Samba is maintained by a dedicated group of people who volunteer their time, skills and efforts. We receive far more mail than we can possibly answer, so you have a much higher chance of a response and a fix if you send us a “*developer friendly*” bug report that lets us fix it fast.

Do not assume that if you post the bug to the `comp.protocols.smb` newsgroup or the mailing list that we will read it. If you suspect that your problem is not a bug but a configuration problem, it is better to send it to the Samba mailing list, as there are thousands of other users on that list who may be able to help you.

You may also like to look though the recent mailing list archives, which are conveniently accessible on the Samba Web pages at <http://samba.org/samba/>.

34.2 General Information

Before submitting a bug report, check your config for silly errors. Look in your log files for obvious messages that tell you've misconfigured something. Run `testparm` to check your config file for correct syntax.

Have you looked through Chapter 32, *The Samba Checklist*? This is extremely important.

If you include part of a log file with your bug report, then be sure to annotate it with exactly what you were doing on the client at the time and exactly what the results were.

34.3 Debug Levels

If the bug has anything to do with Samba behaving incorrectly as a server (like refusing to open a file), then the log files will probably be quite useful. Depending on the problem, a

¹<https://bugzilla.samba.org/>

log level of between 3 and 10 showing the problem may be appropriate. A higher level gives more detail, but may use too much disk space.

To set the debug level, use the *log level* in your `smb.conf`. You may also find it useful to set the log level higher for just one machine and keep separate logs for each machine. To do this, add the following lines to your main `smb.conf` file:

```
log level = 10
log file = /usr/local/samba/lib/log.%m
include = /usr/local/samba/lib/smb.conf.%m
```

and create a file `/usr/local/samba/lib/smb.conf.machine` where *machine* is the name of the client you wish to debug. In that file put any `smb.conf` commands you want, for example *log level* may be useful. This also allows you to experiment with different security systems, protocol levels and so on, on just one machine.

The `smb.conf` entry *log level* is synonymous with the parameter *debuglevel* that has been used in older versions of Samba and is being retained for backward compatibility of `smb.conf` files.

As the *log level* value is increased, you will record a significantly greater level of debugging information. For most debugging operations, you may not need a setting higher than 3. Nearly all bugs can be tracked at a setting of 10, but be prepared for a large volume of log data.

34.4 Internal Errors

If you get the message “*INTERNAL ERROR*” in your log files, it means that Samba got an unexpected signal while running. It is probably a segmentation fault and almost certainly means a bug in Samba (unless you have faulty hardware or system software).

If the message came from `smbd`, it will probably be accompanied by a message that details the last SMB message received by `smbd`. This information is often useful in tracking down the problem so please include it in your bug report.

You should also detail how to reproduce the problem, if possible. Please make this reasonably detailed.

You may also find that a core file appeared in a `corefiles` subdirectory of the directory where you keep your Samba log files. This file is the most useful tool for tracking down the bug. To use it, you do this:

```
$ gdb smbd core
```

adding appropriate paths to `smbd` and `core` so `gdb` can find them. If you do not have `gdb`, try `dbx`. Then within the debugger, use the command **where** to give a stack trace of where the problem occurred. Include this in your report.

If you know any assembly language, do a **disass** of the routine where the problem occurred (if its in a library routine, then disassemble the routine that called it) and try to work out

exactly where the problem is by looking at the surrounding code. Even if you do not know assembly, including this information in the bug report can be useful.

34.5 Attaching to a Running Process

Unfortunately, some UNIXes (in particular some recent Linux kernels) refuse to dump a core file if the task has changed uid (which `smbd` does often). To debug with this sort of system, you could try to attach to the running process using `gdb smbd PID` where you get `PID` from `smbstatus`. Then use `c` to continue and try to cause the core dump using the client. The debugger should catch the fault and tell you where it occurred.

34.6 Patches

The best sort of bug report is one that includes a fix! If you send us patches, please use `diff -u` format if your version of `diff` supports it, otherwise use `diff -c4`. Make sure you do the diff against a clean version of the source and let me know exactly what version you used.

Part VI

Appendixes

HOW TO COMPILE SAMBA

You can obtain the Samba source from the Samba Website.¹ To obtain a development version, you can download Samba from CVS or using **rsync**.

35.1 Access Samba Source Code via CVS

35.1.1 Introduction

Samba is developed in an open environment. Developers use Concurrent Versioning System (CVS) to “*checkin*” (also known as “*commit*”) new source code. Samba’s various CVS branches can be accessed via anonymous CVS using the instructions detailed in this chapter.

This chapter is a modified version of the instructions found at <http://samba.org/samba/cvs.html>

35.1.2 CVS Access to samba.org

The machine samba.org runs a publicly accessible CVS repository for access to the source code of several packages, including Samba, **rsync**, **distcc**, **ccache**, and **jitterbug**. There are two main ways of accessing the CVS server on this host:

35.1.2.1 Access via CVSweb

You can access the source code via your favorite WWW browser. This allows you to access the contents of individual files in the repository and also to look at the revision history and commit logs of individual files. You can also ask for a diff listing between any two versions on the repository.

Use the URL: <http://samba.org/cgi-bin/CVSweb>

¹<http://samba.org/>

35.1.2.2 Access via CVS

You can also access the source code via a normal CVS client. This gives you much more control over what you can do with the repository and allows you to checkout whole source trees and keep them up-to-date via normal CVS commands. This is the preferred method of access if you are a developer and not just a casual browser.

To download the latest CVS source code, point your browser at the URL : <http://www.cyclic.com/>. and click on the “*How to get CVS*” link. CVS is free software under the GNU GPL (as is Samba). Note that there are several graphical CVS clients that provide a graphical interface to the sometimes mundane CVS commands. Links to these clients are also available from the Cyclic Web site.

To gain access via anonymous CVS, use the following steps. For this example it is assumed that you want a copy of the Samba source code. For the other source code repositories on this system just substitute the correct package name.

RETRIEVING SAMBA USING CVS

1. Install a recent copy of CVS. All you really need is a copy of the CVS client binary.
2. Run the command: `cvs -d :pserver:cvs@samba.org:/cvsroot login`
3. When it asks you for a password, type `cvs`.
4. Run the command `cvs -d :pserver:CVS@samba.org:/cvsroot co samba`. This will create a directory called `samba` containing the latest Samba source code (i.e., the HEAD tagged CVS branch). This currently corresponds to the 3.0 development tree. CVS branches other than HEAD can be obtained by using the `-r` and defining a tag name. A list of branch tag names can be found on the “*Development*” page of the Samba Web site. A common request is to obtain the latest 3.0 release code. This could be done by using the following command: `cvs -d :pserver:cvs@samba.org:/cvsroot co -r SAMBA_3_0 samba`.
5. Whenever you want to merge in the latest code changes, use the following command from within the Samba directory: `cvs update -d -P`

35.2 Accessing the Samba Sources via rsync and ftp

`pserver.samba.org` also exports unpacked copies of most parts of the CVS tree at `ftp://pserver.samba.org/pub/unpacked` and also via anonymous rsync at `rsync://pserver.samba.org/ftp/unpacked/`. I recommend using rsync rather than ftp. See the rsync home-page for more info on rsync.

The disadvantage of the unpacked trees is that they do not support automatic merging of local changes like CVS does. `rsync` access is most convenient for an initial install.

35.3 Verifying Samba’s PGP Signature

It is strongly recommended that you verify the PGP signature for any source file before installing it. Even if you’re not downloading from a mirror site, verifying PGP signatures

should be a standard reflex. Many people today use the GNU GPG toolset in place of PGP. GPG can substitute for PGP.

With that said, go ahead and download the following files:

```
$ wget http://us1.samba.org/samba/ftp/samba-2.2.8a.tar.asc
$ wget http://us1.samba.org/samba/ftp/samba-pubkey.asc
```

The first file is the PGP signature for the Samba source file; the other is the Samba public PGP key itself. Import the public PGP key with:

```
$ gpg --import samba-pubkey.asc
```

and verify the Samba source code integrity with:

```
$ gzip -d samba-2.2.8a.tar.gz
$ gpg --verify samba-2.2.8a.tar.asc
```

If you receive a message like, “*Good signature from Samba Distribution Verification Key...*” then all is well. The warnings about trust relationships can be ignored. An example of what you would not want to see would be:

```
gpg: BAD signature from Samba Distribution Verification Key
```

35.4 Building the Binaries

To build the binaries, first run the program `./configure` in the source directory. This should automatically configure Samba for your operating system. If you have unusual needs, then you may wish to run

```
root# ./configure --help
```

first to see what special options you can enable. Now execute `./configure` with any arguments it might need:

```
root# ./configure [... arguments ...]
```

Executing

```
root# make
```

will create the binaries. Once it is successfully compiled you can use

```
root# make install
```

to install the binaries and manual pages. You can separately install the binaries and/or man pages using

```
root# make installbin
```

and

```
root# make installman
```

Note that if you are upgrading from a previous version of Samba you might like to know that the old versions of the binaries will be renamed with an “.old” extension. You can go back to the previous version with

```
root# make revert
```

if you find this version a disaster!

35.4.1 Compiling Samba with Active Directory Support

In order to compile Samba with ADS support, you need to have installed on your system:

- The MIT or Heimdal kerberos development libraries (either install from the sources or use a package).
- The OpenLDAP development libraries.

If your kerberos libraries are in a non-standard location, then remember to add the configure option `--with-krb5=DIR`.

After you run configure, make sure that `include/config.h` it generates contain lines like this:

```
#define HAVE_KRB5 1
#define HAVE_LDAP 1
```

If it does not, configure did not find your KRB5 libraries or your LDAP libraries. Look in `config.log` to figure out why and fix it.

35.4.1.1 Installing the Required Packages for Debian

On Debian, you need to install the following packages:

- libkrb5-dev
- krb5-user

35.4.1.2 Installing the Required Packages for Red Hat Linux

On Red Hat Linux, this means you should have at least:

- `krb5-workstation` (for `kinit`)
- `krb5-libs` (for linking with)
- `krb5-devel` (because you are compiling from source)

in addition to the standard development environment.

If these files are not installed on your system, you should check the installation CDs to find which has them and install the files using your tool of choice. If in doubt about what tool to use, refer to the Red Hat Linux documentation.

35.4.1.3 SuSE Linux Package Requirements

SuSE Linux installs Heimdal packages that may be required to allow you to build binary packages. You should verify that the development libraries have been installed on your system.

SuSE Linux Samba RPMs support Kerberos. Please refer to the documentation for your SuSE Linux system for information regarding SuSE Linux specific configuration. Additionally, SuSE are very active in the maintenance of Samba packages that provide the maximum capabilities that are available. You should consider using SuSE provided packages where they are available.

35.5 Starting the `smbd` and `nmbd`

You must choose to start `smbd` and `nmbd` either as daemons or from `inetd`. Don't try to do both! Either you can put them in `inetd.conf` and have them started on demand by `inetd` or `xinetd`, or you can start them as daemons either from the command line or in `/etc/rc.local`. See the man pages for details on the command line options. Take particular care to read the bit about what user you need to have to start Samba. In many cases, you must be root.

The main advantage of starting `smbd` and `nmbd` using the recommended daemon method is that they will respond slightly more quickly to an initial connection request.

35.5.1 Starting from `inetd.conf`

NOTE



The following will be different if you use NIS, NIS+ or LDAP to distribute services maps.

Look at your `/etc/services`. What is defined at port 139/tcp? If nothing is defined, then add a line like this:

```
netbios-ssn    139/tcp
```

Similarly for 137/udp, you should have an entry like:

```
netbios-ns    137/udp
```

Next, edit your `/etc/inetd.conf` and add two lines like this:

```
netbios-ssn stream tcp nowait root /usr/local/samba/bin/smbd smbd
netbios-ns  dgram  udp  wait  root /usr/local/samba/bin/nmbd nmbd
```

The exact syntax of `/etc/inetd.conf` varies between UNIXes. Look at the other entries in `inetd.conf` for a guide.

Some distributions use `xinetd` instead of `inetd`. Consult the `xinetd` manual for configuration information.

NOTE



Some UNIXes already have entries like `netbios_ns` (note the underscore) in `/etc/services`. You must edit `/etc/services` or `/etc/inetd.conf` to make them consistent.

NOTE



On many systems you may need to use the `interfaces` option in `smb.conf` to specify the IP address and netmask of your interfaces. Run `ifconfig` as root if you do not know what the broadcast is for your net. `nmbd` tries to determine it at run time, but fails on some UNIXes.

WARNING



Many UNIXes only accept around five parameters on the command line in `inetd.conf`. This means you shouldn't use spaces between the options and arguments, or you should use a script and start the script from **`inetd`**.

Restart `inetd`, perhaps just send it a HUP.

```
root# killall -HUP inetd
```

35.5.2 Alternative: Starting `smbd` as a Daemon

To start the server as a daemon, you should create a script something like this one, perhaps calling it `start smb`.

```
#!/bin/sh
/usr/local/samba/bin/smbd -D
/usr/local/samba/bin/nmbd -D
```

Make it executable with `chmod +x start smb`

You can then run `start smb` by hand or execute it from `/etc/rc.local`.

To kill it, send a kill signal to the processes `nmbd` and `smbd`.

NOTE



If you use the SVR4 style init system, you may like to look at the `examples/svr4-startup` script to make Samba fit into that system.

PORTABILITY

Samba works on a wide range of platforms but the interface all the platforms provide is not always compatible. This chapter contains platform-specific information about compiling and using Samba.

36.1 HP-UX

HP's implementation of supplementary groups is non-standard (for historical reasons). There are two group files, `/etc/group` and `/etc/logingroup`; the system maps UIDs to numbers using the former, but `initgroups()` reads the latter. Most system admins who know the ropes symlink `/etc/group` to `/etc/logingroup` (hard link does not work for reasons too obtuse to go into here). `initgroups()` will complain if one of the groups you're in in `/etc/logingroup` has what it considers to be an invalid ID, which means outside the range `[0..UID_MAX]`, where `UID_MAX` is (I think) 60000 currently on HP-UX. This precludes -2 and 65534, the usual `nobody` GIDs.

If you encounter this problem, make sure the programs that are failing to `initgroups()` are run as users, not in any groups with GIDs outside the allowed range.

This is documented in the HP manual pages under `setgroups(2)` and `passwd(4)`.

On HP-UX you must use `gcc` or the HP ANSI compiler. The free compiler that comes with HP-UX is not ANSI compliant and cannot compile Samba.

36.2 SCO UNIX

If you run an old version of SCO UNIX, you may need to get important TCP/IP patches for Samba to work correctly. Without the patch, you may encounter corrupt data transfers using Samba.

The patch you need is UOD385 Connection Drivers SLS. It is available from SCO (<ftp.sco.com>, directory SLS, files `uod385a.Z` and `uod385a.ltr.Z`).

The information provided here refers to an old version of SCO UNIX. If you require binaries for more recent SCO UNIX products, please contact SCO to obtain packages that are ready to install. You should also verify with SCO that your platform is up-to-date for the binary

packages you will install. This is important if you wish to avoid data corruption problems with your installation. To build Samba for SCO UNIX products may require significant patching of Samba source code. It is much easier to obtain binary packages directly from SCO.

36.3 DNIX

DNIX has a problem with `seteuid()` and `setegid()`. These routines are needed for Samba to work correctly, but they were left out of the DNIX C library for some reason.

For this reason Samba by default defines the macro `NO_EID` in the DNIX section of `includes.h`. This works around the problem in a limited way, but it is far from ideal, and some things still will not work right.

To fix the problem properly, you need to assemble the following two functions and then either add them to your C library or link them into Samba. Put the following in the file `setegid.s`:

```

        .globl  _setegid
_setegid:
        moveq   #47,d0
        movl    #100,a0
        moveq   #1,d1
        movl    4(sp),a1
        trap    #9
        bccs   1$
        jmp     cerror
1$:
        clrl   d0
        rts

```

Put this in the file `seteuid.s`:

```

        .globl  _seteuid
_seteuid:
        moveq   #47,d0
        movl    #100,a0
        moveq   #0,d1
        movl    4(sp),a1
        trap    #9
        bccs   1$
        jmp     cerror
1$:
        clrl   d0
        rts

```


After creating the above files, you then assemble them using

```
$ as seteuid.s
$ as setegid.s
```

that should produce the files `seteuid.o` and `setegid.o`

Then you need to add these to the LIBSM line in the DNIX section of the Samba Makefile. Your LIBSM line will then look something like this:

```
LIBSM = setegid.o seteuid.o -ln
```

You should then remove the line:

```
#define NO_EID
```

from the DNIX section of `includes.h`.

36.4 Red Hat Linux

By default during installation, some versions of Red Hat Linux add an entry to `/etc/hosts` as follows:

```
127.0.0.1 loopback "hostname"."domainname"
```

This causes Samba to loop back onto the loopback interface. The result is that Samba fails to communicate correctly with the world and therefore may fail to correctly negotiate who is the master browse list holder and who is the master browser.

Corrective Action: Delete the entry after the word "loopback" in the line starting 127.0.0.1.

36.5 AIX

36.5.1 Sequential Read Ahead

Disabling Sequential Read Ahead using `vmtune -r 0` improves Samba performance significantly.

36.6 Solaris

36.6.1 Locking Improvements

Some people have been experiencing problems with `F_SETLKW64/fcntl` when running Samba on Solaris. The built-in file locking mechanism was not scalable. Performance would degrade to the point where processes would get into loops of trying to lock a file. It would try a lock, then fail, then try again. The lock attempt was failing before the grant was occurring. So the visible manifestation of this would be a handful of processes stealing all of the CPU, and when they were trussed they would be stuck if `F_SETLKW64` loops.

Sun released patches for Solaris 2.6, 8, and 9. The patch for Solaris 7 has not been released yet.

The patch revision for 2.6 is 105181-34, for 8 is 108528-19 and for 9 is 112233-04.

After the install of these patches, it is recommended to reconfigure and rebuild Samba.

Thanks to Joe Meslovich for reporting this.

36.6.2 Winbind on Solaris 9

Nsswitch on Solaris 9 refuses to use the Winbind NSS module. This behavior is fixed by Sun in patch 113476-05, which as of March 2003, is not in any roll-up packages.

SAMBA AND OTHER CIFS CLIENTS

This chapter contains client-specific information.

37.1 Macintosh Clients

Yes. Thursby¹ has a CIFS Client/Server called DAVE.² They test it against Windows 95, Windows NT /200x/XP and Samba for compatibility issues. At the time of this writing, DAVE was at version 4.1. Please refer to Thursby's Web site for more information regarding this product.

Alternatives — There are two free implementations of AppleTalk for several kinds of UNIX machines and several more commercial ones. These products allow you to run file services and print services natively to Macintosh users, with no additional support required on the Macintosh. The two free implementations are Netatalk,³ and CAP.⁴ What Samba offers MS Windows users, these packages offer to Macs. For more info on these packages, Samba, and Linux (and other UNIX-based systems), see http://www.eats.com/linux_mac_win.html.

Newer versions of the Macintosh (Mac OS X) include Samba.

37.2 OS2 Client

37.2.1 Configuring OS/2 Warp Connect or OS/2 Warp 4

Basically, you need three components:

- The File and Print Client (IBM Peer)
- TCP/IP (Internet support)
- The “*NetBIOS over TCP/IP*” driver (TCPBEUI)

¹<http://www.thursby.com/>

²<http://www.thursby.com/products/dave.html>

³<http://www.umich.edu/~rsug/netatalk/>

⁴<http://www.cs.mu.oz.au/appletalk/atalk.html>

Installing the first two together with the base operating system on a blank system is explained in the Warp manual. If Warp has already been installed, but you now want to install the networking support, use the “*Selective Install for Networking*” object in the “*System Setup*” folder.

Adding the “*NetBIOS over TCP/IP*” driver is not described in the manual and just barely in the online documentation. Start **MPTS.EXE**, click on **OK**, click on **Configure LAPS** and click on **IBM OS/2 NETBIOS OVER TCP/IP** in **Protocols**. This line is then moved to **Current Configuration**. Select that line, click on **Change number** and increase it from 0 to 1. Save this configuration.

If the Samba server is not on your local subnet, you can optionally add IP names and addresses of these servers to the **Names List**, or specify a WINS server (NetBIOS Nameserver in IBM and RFC terminology). For Warp Connect, you may need to download an update for IBM Peer to bring it on the same level as Warp 4. See the Web page mentioned above.

37.2.2 Configuring Other Versions of OS/2

This sections deals with configuring OS/2 Warp 3 (not Connect), OS/2 1.2, 1.3 or 2.x.

You can use the free Microsoft LAN Manager 2.2c Client for OS/2 that is available from <ftp://ftp.microsoft.com/BusSys/Clients/LANMAN.OS2/>. In a nutshell, edit the file `\OS2VER` in the root directory of the OS/2 boot partition and add the lines:

```
20=setup.exe
20=netwksta.sys
20=netvdd.sys
```

before you install the client. Also, do not use the included NE2000 driver because it is buggy. Try the NE2000 or NS2000 driver from <ftp://ftp.cdrom.com/pub/os2/network/ndis/> instead.

37.2.3 Printer Driver Download for OS/2 Clients

Create a share called `[PRINTDRV]` that is world-readable. Copy your OS/2 driver files there. The `.EA_` files must still be separate, so you will need to use the original install files and not copy an installed driver from an OS/2 system.

Install the NT driver first for that printer. Then, add to your `smb.conf` a parameter, `os2 driver map = filename`. Next, in the file specified by `filename`, map the name of the NT driver name to the OS/2 driver name as follows:

```
nt driver name = os2 driver name.device name, e.g.
```

```
HP LaserJet 5L = LASERJET.HP LaserJet 5L
```

You can have multiple drivers mapped in this file.

If you only specify the OS/2 driver name, and not the device name, the first attempt to download the driver will actually download the files, but the OS/2 client will tell you the

driver is not available. On the second attempt, it will work. This is fixed simply by adding the device name to the mapping, after which it will work on the first attempt.

37.3 Windows for Workgroups

37.3.1 Latest TCP/IP Stack from Microsoft

Use the latest TCP/IP stack from Microsoft if you use Windows for Workgroups. The early TCP/IP stacks had lots of bugs.

Microsoft has released an incremental upgrade to their TCP/IP 32-bit VxD drivers. The latest release can be found on their ftp site at ftp.microsoft.com, located in /peropsys/windows/public/tcpip/wfwt32.exe. There is an update.txt file there that describes the problems that were fixed. New files include WINSOCK.DLL, TELNET.EXE, WSOCK.386, VNBT.386, WSTCP.386, TRACERT.EXE, NETSTAT.EXE, and NBTSTAT.EXE.

37.3.2 Delete .pwl Files After Password Change

Windows for Workgroups does a lousy job with passwords. When you change passwords on either the UNIX box or the PC, the safest thing to do is to delete the .pwl files in the Windows directory. The PC will complain about not finding the files, but will soon get over it, allowing you to enter the new password.

If you do not do this, you may find that Windows for Workgroups remembers and uses the old password, even if you told it a new one.

Often Windows for Workgroups will totally ignore a password you give it in a dialog box.

37.3.3 Configuring Windows for Workgroups Password Handling

There is a program call `admincfg.exe` on the last disk (disk 8) of the WFW 3.11 disk set. To install it, type `EXPAND A:\ADMINCFG.EX_ C:\WINDOWS\ADMINCFG.EXE`. Then add an icon for it via the Program Manager **New** Menu. This program allows you to control how WFW handles passwords, i.e., Disable Password Caching and so on. for use with *security* = user.

37.3.4 Password Case Sensitivity

Windows for Workgroups upcases the password before sending it to the server. UNIX passwords can be case-sensitive though. Check the `smb.conf` information on *password level* to specify what characters Samba should try to upcase when checking.

37.3.5 Use TCP/IP as Default Protocol

To support print queue reporting, you may find that you have to use TCP/IP as the default protocol under Windows for Workgroups. For some reason, if you leave NetBEUI as the

default, it may break the print queue reporting on some systems. It is presumably a Windows for Workgroups bug.

37.3.6 Speed Improvement

Note that some people have found that setting *DefaultRcvWindow* in the *[MSTCP]* section of the *SYSTEM.INI* file under Windows for Workgroups to 3072 gives a big improvement.

My own experience with *DefaultRcvWindow* is that I get a much better performance with a large value (16384 or larger). Other people have reported that anything over 3072 slows things down enormously. One person even reported a speed drop of a factor of 30 when he went from 3072 to 8192.

37.4 Windows 95/98

When using Windows 95 OEM SR2, the following updates are recommended where Samba is being used. Please note that the above change will effect you once these updates have been installed.

There are more updates than the ones mentioned here. You are referred to the Microsoft Web site for all currently available updates to your specific version of Windows 95.

Kernel Update: KRNLUPD.EXE
Ping Fix: PINGUPD.EXE
RPC Update: RPCRTUPD.EXE
TCP/IP Update: VIPUPD.EXE
Redirector Update: VRDRUPD.EXE

Also, if using MS Outlook, it is desirable to install the **OLEUPD.EXE** fix. This fix may stop your machine from hanging for an extended period when exiting Outlook and you may notice a significant speedup when accessing network neighborhood services.

37.4.1 Speed Improvement

Configure the Windows 95 TCP/IP registry settings to give better performance. I use a program called **MTUSPEED.exe** that I got off the Internet. There are various other utilities of this type freely available.

37.5 Windows 2000 Service Pack 2

There are several annoyances with Windows 2000 SP2. One of which only appears when using a Samba server to host user profiles to Windows 2000 SP2 clients in a Windows domain. This assumes that Samba is a member of the domain, but the problem will most likely occur if it is not.

In order to serve profiles successfully to Windows 2000 SP2 clients (when not operating as a PDC), Samba must have *nt acl support* = no added to the file share which houses the roaming profiles. If this is not done, then the Windows 2000 SP2 client will complain about

not being able to access the profile (Access Denied) and create multiple copies of it on disk (DOMAIN.user.001, DOMAIN.user.002, and so on). See the `smb.conf` man page for more details on this option. Also note that the `nt acl support` parameter was formally a global parameter in releases prior to Samba 2.2.2.

Example 37.1 provides a minimal profile share.

Example 37.1. Minimal profile share

```
[profile]
path = /export/profile
create mask = 0600
directory mask = 0700
nt acl support = no
read only = no
```

The reason for this bug is that the Windows 200x SP2 client copies the security descriptor for the profile that contains the Samba server's SID, and not the domain SID. The client compares the SID for SAMBA\user and realizes it is different from the one assigned to DOMAIN\user. Hence, the reason for the access denied message.

By disabling the `nt acl support` parameter, Samba will send the Windows 200x client a response to the QuerySecurityDescriptor trans2 call, which causes the client to set a default ACL for the profile. This default ACL includes:

```
DOMAIN\user "Full Control">
```

NOTE



This bug does not occur when using Winbind to create accounts on the Samba host for Domain users.

37.6 Windows NT 3.1

If you have problems communicating across routers with Windows NT 3.1 workstations, read this Microsoft Knowledge Base article.⁵

⁵<http://support.microsoft.com/default.aspx?scid=kb;Q103765>

SAMBA PERFORMANCE TUNING

38.1 Comparisons

The Samba server uses TCP to talk to the client. Thus if you are trying to see if it performs well, you should really compare it to programs that use the same protocol. The most readily available programs for file transfer that use TCP are ftp or another TCP-based SMB server.

If you want to test against something like an NT or Windows for Workgroups server, then you will have to disable all but TCP on either the client or server. Otherwise, you may well be using a totally different protocol (such as NetBEUI) and comparisons may not be valid.

Generally, you should find that Samba performs similarly to ftp at raw transfer speed. It should perform quite a bit faster than NFS, although this depends on your system.

Several people have done comparisons between Samba and Novell, NFS or Windows NT. In some cases Samba performed the best, in others the worst. I suspect the biggest factor is not Samba versus some other system, but the hardware and drivers used on the various systems. Given similar hardware, Samba should certainly be competitive in speed with other systems.

38.2 Socket Options

There are a number of socket options that can greatly affect the performance of a TCP-based server like Samba.

The socket options that Samba uses are settable both on the command line with the `-O` option, or in the `smb.conf` file.

The *socket options* section of the `smb.conf` manual page describes how to set these and gives recommendations.

Getting the socket options correct can make a big difference to your performance, but getting them wrong can degrade it by just as much. The correct settings are very dependent on your local network.

The socket option `TCP_NODELAY` is the one that seems to make the biggest single difference for most networks. Many people report that adding `socket options = TCP_NODELAY` doubles the read performance of a Samba drive. The best explanation I have seen for this is that the Microsoft TCP/IP stack is slow in sending TCP ACKs.

There have been reports that setting `socket options = SO_RCVBUF=8192` in `smb.conf` can seriously degrade Samba performance on the loopback adaptor (IP Address 127.0.0.1). It is strongly recommended that before specifying any settings for `socket options` the effect first be quantitatively measured on the server being configured.

38.3 Read Size

The option `read size` affects the overlap of disk reads/writes with network reads/writes. If the amount of data being transferred in several of the SMB commands (currently SMBwrite, SMBwriteX and SMBreadbrow) is larger than this value, then the server begins writing the data before it has received the whole packet from the network, or in the case of SMBreadbrow, it begins writing to the network before all the data has been read from disk.

This overlapping works best when the speeds of disk and network access are similar, having little effect when the speed of one is much greater than the other.

The default value is 16384, but little experimentation has been done as yet to determine the optimal value, and it is likely that the best value will vary greatly between systems anyway. A value over 65536 is pointless and will cause you to allocate memory unnecessarily.

38.4 Max Xmit

At startup the client and server negotiate a *maximum transmit* size, which limits the size of nearly all SMB commands. You can set the maximum size that Samba will negotiate using the `max xmit` option in `smb.conf`. Note that this is the maximum size of SMB requests that Samba will accept, but not the maximum size that the client will accept. The client maximum receive size is sent to Samba by the client and Samba honors this limit.

It defaults to 65536 bytes (the maximum), but it is possible that some clients may perform better with a smaller transmit unit. Trying values of less than 2048 is likely to cause severe problems. In most cases the default is the best option.

38.5 Log Level

If you set the log level (also known as *debug level*) higher than 2 then you may suffer a large drop in performance. This is because the server flushes the log file after each operation, which can be quite expensive.

38.6 Read Raw

The *read raw* operation is designed to be an optimized, low-latency file read operation. A server may choose to not support it, however, and Samba makes support for *read raw* optional, with it being enabled by default.

In some cases clients do not handle *read raw* very well and actually get lower performance using it than they get using the conventional read operations.

So you might like to try *read raw = no* and see what happens on your network. It might lower, raise or not effect your performance. Only testing can really tell.

38.7 Write Raw

The *write raw* operation is designed to be an optimized, low-latency file write operation. A server may choose to not support it, however, and Samba makes support for *write raw* optional, with it being enabled by default.

Some machines may find *write raw* slower than normal write, in which case you may wish to change this option.

38.8 Slow Logins

Slow logins are almost always due to the password checking time. Using the lowest practical *password level* will improve things.

38.9 Client Tuning

Often a speed problem can be traced to the client. The client (for example Windows for Workgroups) can often be tuned for better TCP performance. Check the sections on the various clients in Chapter 37, *Samba and Other CIFS Clients*.

38.10 Samba Performance Problem Due to Changing Linux Kernel

A user wrote the following to the mailing list:

I am running Gentoo on my server and Samba 2.2.8a. Recently I changed kernel version from `linux-2.4.19-gentoo-r10` to `linux-2.4.20-wolk4.0s`. And now I have a performance issue with Samba. Many of you will probably say, “*Move to vanilla sources!*” Well, I tried that and it didn’t work. I have a 100mb LAN and two computers (Linux and Windows 2000). The Linux server shares directories with DivX files, the client (Windows 2000) plays them via LAN. Before when I was running the 2.4.19 kernel everything was fine, but now movies freeze and stop. I tried moving files between the server and Windows and it is terribly slow.

The answer he was given is:

Grab the mii-tool and check the duplex settings on the NIC. My guess is that it is a link layer issue, not an application layer problem. Also run ifconfig and verify that the framing error, collisions, and so on, look normal for ethernet.

38.11 Corrupt tdb Files

Our Samba PDC server has been hosting three TB of data to our 500+ users [Windows NT/XP] for the last three years using Samba without a problem. Today all shares went very slow. Also the main smbd kept spawning new processes so we had 1600+ running smbd's (normally we avg. 250). It crashed the SUN E3500 cluster twice. After a lot of searching, I decided to **rm /var/locks/*.tdb**. Happy again.

Question: Is there any method of keeping the *.tdb files in top condition or how can I detect early corruption?

Answer: Yes, run **tdbbackup** each time after stopping nmbd and before starting nmbd.

Question: What I also would like to mention is that the service latency seems a lot lower than before the locks cleanup. Any ideas on keeping it top notch?

Answer: Yes. Same answer as for previous question!

38.12 Samba Performance is Very Slow

A site reported experiencing very baffling symptoms with MYOB Premier opening and accessing it's datafiles. Some operations on the file would take between 40 and 45 seconds.

It turned out that the printer monitor program running on the windows clients was causing the problems. From the logs, we saw activity coming through with pauses of about 1 second.

Stopping the monitor software resulted in the networks access at normal (quick) speed. Restarting the program caused the speed to slow down again. The printer was a cannon lbp810 and the relevant task was something like CAPON (not sure on spelling). The monitor software displayed a printing now dialog on the client during printing.

We discovered this by starting with a clean install of windows and trying the app at every step of the installation of other software process (had to do this many times).

Moral of the story, check everything (other software included)!

DNS AND DHCP CONFIGURATION GUIDE

39.1 Features and Benefits

There are few subjects in the UNIX world that might raise as much contention as Domain Name System (DNS) and Dynamic Host Configuration Protocol (DHCP). Not all opinions held for or against particular implementations of DNS and DHCP are valid.

We live in a modern age where many information technology users demand mobility and freedom. Microsoft Windows users in particular expect to be able to plug their notebook computer into a network port and have things “*just work.*”

UNIX administrators have a point. Many of the normative practices in the Microsoft Windows world at best border on bad practice from a security perspective. Microsoft Windows networking protocols allow workstations to arbitrarily register themselves on a network. Windows 2000 Active Directory registers entries in the DNS name space that are equally perplexing to UNIX administrators. Welcome to the new world!

The purpose of this chapter is to demonstrate the configuration of the Internet Software Consortiums (ISC) DNS and DHCP servers to provide dynamic services that are compatible with their equivalents in the Microsoft Windows 2000 Server products.

The purpose of this chapter is to provide no more than a working example of configuration files for both DNS and DHCP servers. The examples used match configuration examples used elsewhere in this document.

This chapter explicitly does not provide a tutorial, nor does it pretend to be a reference guide on DNS and DHCP, as this is well beyond the scope and intent of this document as a whole. Anyone who wants more detailed reference materials on DNS or DHCP should visit the ISC Web sites at <http://www.isc.org>. Those wanting a written text might also be interested in the O'Reilly publications on these two subjects.

39.2 Example Configuration

The domain name system is to the Internet what water is to life. By it nearly all information resources (host names) are resolved to their Internet protocol (IP) address. Windows

networking tried hard to avoid the complexities of DNS, but alas, DNS won. The alternative to DNS, the Windows Internet Name Service (WINS) an artifact of NetBIOS networking over the TCP/IP protocols, has demonstrated scalability problems as well as a flat non-hierarchical name space that became unmanageable as the size and complexity of information technology networks grew.

WINS is a Microsoft implementation of the RFC1001/1002 NetBIOS Name Service (NBNS). It allows NetBIOS clients (like Microsoft Windows Machines) to register an arbitrary machine name that the administrator or user has chosen together with the IP address that the machine has been given. Through the use of WINS, network client machines could resolve machine names to their IP address.

The demand for an alternative to the limitations of NetBIOS networking finally drove Microsoft to use DNS and Active Directory. Microsoft's new implementation attempts to use DNS in a manner similar to the way that WINS is used for NetBIOS networking. Both WINS and Microsoft DNS rely on dynamic name registration.

Microsoft Windows clients can perform dynamic name registration to the DNS server on start-up. Alternately, where DHCP is used to assign workstation IP addresses, it is possible to register host names and their IP address by the DHCP server as soon as a client acknowledges an IP address lease. Lastly, Microsoft DNS can resolve hostnames via Microsoft WINS.

The following configurations demonstrate a simple insecure Dynamic DNS server and a simple DHCP server that matches the DNS configuration.

39.2.1 Dynamic DNS

The example DNS configuration is for a private network in the IP address space for network 192.168.1.0/24. The private class network address space is set forth in RFC1918.

It is assumed that this network will be situated behind a secure firewall. The files that follow work with ISC BIND version 9. BIND is the Berkely Internet Name Daemon. The following configuration files are offered:

The master configuration file for `/etc/named.conf` determines the location of all further configuration files used. The location and name of this file is specified in the start-up script that is part of the operating system.

```
# Quenya.Org configuration file
```

```
acl mynet {
    192.168.1.0/24;
    127.0.0.1;
};

options {
    directory "/var/named";
    listen-on-v6 { any; };
};
```

```
notify no;
forward first;
forwarders {
    192.168.1.1;
};
auth-nxdomain yes;
multiple-cnames yes;
listen-on {
    mynet;
};
};
```

```
# The following three zone definitions do not need any modification.
# The first one defines localhost while the second defines the
# reverse lookup for localhost. The last zone "." is the
# definition of the root name servers.
```

```
zone "localhost" in {
    type master;
    file "localhost.zone";
};
```

```
zone "0.0.127.in-addr.arpa" in {
    type master;
    file "127.0.0.zone";
};
```

```
zone "." in {
    type hint;
    file "root.hint";
};
```

```
# You can insert further zone records for your own domains below.
```

```
zone "kenya.org" {
    type master;
    file "/var/named/kenya.org.hosts";
    allow-query {
        mynet;
    };
    allow-transfer {
        mynet;
    };
    allow-update {
        mynet;
    };
};
```

```

zone "1.168.192.in-addr.arpa" {
    type master;
    file "/var/named/192.168.1.0.rev";
    allow-query {
        mynet;
    };
    allow-transfer {
        mynet;
    };
    allow-update {
        mynet;
    };
};

```

The following files are all located in the directory `/var/named`. This is the `/var/named/localhost.zone` file:

```

$TTL 1W
@           IN SOA  @       root (
            42          ; serial (d. adams)
            2D          ; refresh
            4H          ; retry
            6W          ; expiry
            1W )        ; minimum

            IN NS      @
            IN A       127.0.0.1

```

The `/var/named/127.0.0.zone` file:

```

$TTL 1W
@           IN SOA      localhost. root.localhost. (
            42          ; serial (d. adams)
            2D          ; refresh
            4H          ; retry
            6W          ; expiry
            1W )        ; minimum

            IN NS      localhost.
1           IN PTR     localhost.

```

The `/var/named/kenya.org.host` file:

```

$ORIGIN .
$TTL 38400          ; 10 hours 40 minutes

```



```

quenya.org      IN SOA  marvel.quenya.org. root.quenya.org. (
                2003021832 ; serial
                10800      ; refresh (3 hours)
                3600       ; retry (1 hour)
                604800     ; expire (1 week)
                38400      ; minimum (10 hours 40 minutes)
                )
                NS       marvel.quenya.org.
                MX       10 mail.quenya.org.
$ORIGIN quenya.org.
frodo           A       192.168.1.1
marvel          A       192.168.1.2
;
mail            CNAME   marvel
www             CNAME   marvel

```

The /var/named/192.168.1.0.rev file:

```

$ORIGIN .
$TTL 38400      ; 10 hours 40 minutes
1.168.192.in-addr.arpa IN SOA  marvel.quenya.org. root.quenya.org. (
                2003021824 ; serial
                10800      ; refresh (3 hours)
                3600       ; retry (1 hour)
                604800     ; expire (1 week)
                38400      ; minimum (10 hours 40 minutes)
                )
                NS       marvel.quenya.org.
$ORIGIN 1.168.192.in-addr.arpa.
1           PTR    frodo.quenya.org.
2           PTR    marvel.quenya.org.

```

The above were copied from a fully working system. All dynamically registered entries have been removed. In addition to these files, BIND version 9 will create for each of the dynamic registration files a file that has a .jnl extension. Do not edit or tamper with the configuration files or with the .jnl files that are created.

39.2.2 DHCP Server

The following file is used with the ISC DHCP Server version 3. The file is located in /etc/dhcpd.conf:

```

ddns-updates on;
ddns-domainname "quenya.org";
option ntp-servers 192.168.1.2;

```

```
ddns-update-style ad-hoc;
allow unknown-clients;
default-lease-time 86400;
max-lease-time 172800;

option domain-name "kenya.org";
option domain-name-servers 192.168.1.2;
option netbios-name-servers 192.168.1.2;
option netbios-dd-server 192.168.1.2;
option netbios-node-type 8;

subnet 192.168.1.0 netmask 255.255.255.0 {
    range dynamic-bootp 192.168.1.60 192.168.1.254;
    option subnet-mask 255.255.255.0;
    option routers 192.168.1.2;
    allow unknown-clients;
}
```

In the above example, IP addresses between 192.168.1.1 and 192.168.1.59 are reserved for fixed address (commonly called **hard-wired**) IP addresses. The addresses between 192.168.1.60 and 192.168.1.254 are allocated for dynamic use.

MANUAL PAGES

This appendix contains most of the manual pages from the official Samba distribution. All manual pages have been written by members of the Samba Team¹.

A.1 smb.conf

SYNOPSIS

The `smb.conf` file is a configuration file for the Samba suite. `smb.conf` contains runtime configuration information for the Samba programs. The `smb.conf` file is designed to be configured and administered by the `swat(8)` program. The complete description of the file format and possible parameters held within are here for reference purposes.

FILE FORMAT

The file consists of sections and parameters. A section begins with the name of the section in square brackets and continues until the next section begins. Sections contain parameters of the form

name = value

The file is line-based - that is, each newline-terminated line represents either a comment, a section name or a parameter.

Section and parameter names are not case sensitive.

Only the first equals sign in a parameter is significant. Whitespace before or after the first equals sign is discarded. Leading, trailing and internal whitespace in section and parameter names is irrelevant. Leading and trailing whitespace in a parameter value is discarded. Internal whitespace within a parameter value is retained verbatim.

Any line beginning with a semicolon (“;”) or a hash (“#”) character is ignored, as are lines containing only whitespace.

Any line ending in a “\” is continued on the next line in the customary UNIX fashion.

¹<http://samba.org/samba/team.html>

The values following the equals sign in parameters are all either a string (no quotes needed) or a boolean, which may be given as yes/no, 0/1 or true/false. Case is not significant in boolean values, but is preserved in string values. Some items such as create modes are numeric.

SECTION DESCRIPTIONS

Each section in the configuration file (except for the [global] section) describes a shared resource (known as a “*share*”). The section name is the name of the shared resource and the parameters within the section define the shares attributes.

There are three special sections, [global], [homes] and [printers], which are described under *special sections*. The following notes apply to ordinary section descriptions.

A share consists of a directory to which access is being given plus a description of the access rights which are granted to the user of the service. Some housekeeping options are also specifiable.

Sections are either file share services (used by the client as an extension of their native file systems) or printable services (used by the client to access print services on the host running the server).

Sections may be designated *guest* services, in which case no password is required to access them. A specified UNIX *guest account* is used to define access privileges in this case.

Sections other than guest services will require a password to access them. The client provides the username. As older clients only provide passwords and not usernames, you may specify a list of usernames to check against the password using the “*user =*” option in the share definition. For modern clients such as Windows 95/98/ME/NT/2000, this should not be necessary.

The access rights granted by the server are masked by the access rights granted to the specified or guest UNIX user by the host system. The server does not grant more access than the host system grants.

The following sample section defines a file space share. The user has write access to the path `/home/bar`. The share is accessed via the share name “*foo*”:

Example A.1.

```
[foo]
path = /home/bar
read only = read only = no
```

The following sample section defines a printable share. The share is read-only, but printable. That is, the only write access permitted is via calls to open, write to and close a spool file. The *guest ok* parameter means access will be permitted as the default guest user (specified elsewhere):

Example A.2.

```
[aprinter]
path = /usr/spool/public
read only = yes
printable = yes
guest ok = yes
```

SPECIAL SECTIONS

The [global] section

Parameters in this section apply to the server as a whole, or are defaults for sections that do not specifically define certain items. See the notes under PARAMETERS for more information.

The [homes] section

If a section called [homes] is included in the configuration file, services connecting clients to their home directories can be created on the fly by the server.

When the connection request is made, the existing sections are scanned. If a match is found, it is used. If no match is found, the requested section name is treated as a username and looked up in the local password file. If the name exists and the correct password has been given, a share is created by cloning the [homes] section.

Some modifications are then made to the newly created share:

- The share name is changed from homes to the located username.
- If no path was given, the path is set to the user's home directory.

If you decide to use a *path =* line in your [homes] section, you may find it useful to use the %S macro. For example :

```
path = /data/pchome/%S
```

is useful if you have different home directories for your PCs than for UNIX access.

This is a fast and simple way to give a large number of clients access to their home directories with a minimum of fuss.

A similar process occurs if the requested section name is “homes”, except that the share name is not changed to that of the requesting user. This method of using the [homes] section works well if different users share a client PC.

The [homes] section can specify all the parameters a normal service section can specify, though some make more sense than others. The following is a typical and suitable [homes] section:

Example A.3.

```
[homes]
read only = no
```

An important point is that if guest access is specified in the [homes] section, all home directories will be visible to all clients *without a password*. In the very unlikely event that this is actually desirable, it is wise to also specify *read only access*.

The *browseable* flag for auto home directories will be inherited from the global *browseable* flag, not the [homes] *browseable* flag. This is useful as it means setting *browseable = no* in the [homes] section will hide the [homes] share but make any auto home directories visible.

The [printers] section

This section works like [homes], but for printers.

If a [printers] section occurs in the configuration file, users are able to connect to any printer specified in the local host's printcap file.

When a connection request is made, the existing sections are scanned. If a match is found, it is used. If no match is found, but a [homes] section exists, it is used as described above. Otherwise, the requested section name is treated as a printer name and the appropriate printcap file is scanned to see if the requested section name is a valid printer share name. If a match is found, a new printer share is created by cloning the [printers] section.

A few modifications are then made to the newly created share:

- The share name is set to the located printer name
- If no printer name was given, the printer name is set to the located printer name
- If the share does not permit guest access and no username was given, the username is set to the located printer name.

The [printers] service **MUST** be printable - if you specify otherwise, the server will refuse to load the configuration file.

Typically the path specified is that of a world-writable spool directory with the sticky bit set on it. A typical [printers] entry looks like this:

Example A.4.

```
[printers]
path = /usr/spool/public
guest ok = yes
printable = yes
```

All aliases given for a printer in the `printcap` file are legitimate printer names as far as the server is concerned. If your printing subsystem doesn't work like that, you will have to set up a pseudo-`printcap`. This is a file consisting of one or more lines like this:

```
alias|alias|alias|alias...
```

Each alias should be an acceptable printer name for your printing subsystem. In the `[global]` section, specify the new file as your `printcap`. The server will only recognize names found in your pseudo-`printcap`, which of course can contain whatever aliases you like. The same technique could be used simply to limit access to a subset of your local printers.

An alias, by the way, is defined as any component of the first entry of a `printcap` record. Records are separated by newlines, components (if there are more than one) are separated by vertical bar symbols (“|”).

NOTE



On SYSV systems which use `lpstat` to determine what printers are defined on the system you may be able to use “`printcap name = lpstat`” to automatically obtain a list of printers. See the “`printcap name`” option for more details.

PARAMETERS

Parameters define the specific attributes of sections.

Some parameters are specific to the `[global]` section (e.g., *security*). Some parameters are usable in all sections (e.g., *create mode*). All others are permissible only in normal sections. For the purposes of the following descriptions the `[homes]` and `[printers]` sections will be considered normal. The letter *G* in parentheses indicates that a parameter is specific to the `[global]` section. The letter *S* indicates that a parameter can be specified in a service specific section. All *S* parameters can also be specified in the `[global]` section - in which case they will define the default behavior for all services.

Parameters are arranged here in alphabetical order - this may not create best bedfellows, but at least you can find them! Where there are synonyms, the preferred synonym is described, others refer to the preferred synonym.

VARIABLE SUBSTITUTIONS

Many of the strings that are settable in the config file can take substitutions. For example the option “`path = /tmp/%u`” is interpreted as “`path = /tmp/john`” if the user connected with the username `john`.

These substitutions are mostly noted in the descriptions below, but there are some general substitutions which apply whenever they might be relevant. These are:

%U — session username (the username that the client wanted, not necessarily the same as the one they got).

%G — primary group name of %U.

%h — the Internet hostname that Samba is running on.

%m — the NetBIOS name of the client machine (very useful).

%L — the NetBIOS name of the server. This allows you to change your config based on what the client calls you. Your server can have a “*dual personality*”.

This parameter is not available when Samba listens on port 445, as clients no longer send this information.

%M — the Internet name of the client machine.

%R — the selected protocol level after protocol negotiation. It can be one of CORE, COREPLUS, LANMAN1, LANMAN2 or NT1.

%d — The process id of the current server process.

%a — the architecture of the remote machine. Only some are recognized, and those may not be 100% reliable. It currently recognizes Samba, Windows for Workgroups, Windows 95, Windows NT and Windows 2000. Anything else will be known as “*UNKNOWN*”. If it gets it wrong sending a level 3 log to samba@samba.org² should allow it to be fixed.

%I — The IP address of the client machine.

%T — the current date and time.

%D — Name of the domain or workgroup of the current user.

%(*envvar*) — The value of the environment variable *envvar*.

The following substitutes apply only to some configuration options (only those that are used when a connection has been established):

²<mailto:samba@samba.org>

%S — the name of the current service, if any.

%P — the root directory of the current service, if any.

%u — username of the current service, if any.

%g — primary group name of %u.

%H — the home directory of the user given by %u.

%N — the name of your NIS home directory server. This is obtained from your NIS auto.map entry. If you have not compiled Samba with the *-with-automount* option, this value will be the same as %L.

%p — the path of the service's home directory, obtained from your NIS auto.map entry. The NIS auto.map entry is split up as “%N:%p”.

There are some quite creative things that can be done with these substitutions and other smb.conf options.

NAME MANGLING

Samba supports “*name mangling*” so that DOS and Windows clients can use files that don't conform to the 8.3 format. It can also be set to adjust the case of 8.3 format filenames.

There are several options that control the way mangling is performed, and they are grouped here rather than listed separately. For the defaults look at the output of the testparm program.

All of these options can be set separately for each service (or globally, of course).

The options are:

mangle case = yes/no — controls whether names that have characters that aren't of the “*default*” case are mangled. For example, if this is yes, a name like “*Mai!*” will be mangled. Default *no*.

case sensitive = yes/no — controls whether filenames are case sensitive. If they aren't, Samba must do a filename search and match on passed names. Default *no*.

default case = upper/lower — controls what the default case is for new filenames. Default *lower*.

preserve case = yes/no — controls whether new files are created with the case that the client passes, or if they are forced to be the “*default*” case. Default *yes*.

short preserve case = yes/no — controls if new files which conform to 8.3 syntax, that is all in upper case and of suitable length, are created upper case, or if they are forced to be the “*default*” case. This option can be used with “*preserve case = yes*” to permit long filenames to retain their case, while short names are lowercased. Default *yes*.

By default, Samba 3.0 has the same semantics as a Windows NT server, in that it is case insensitive but case preserving.

NOTE ABOUT USERNAME/PASSWORD VALIDATION

There are a number of ways in which a user can connect to a service. The server uses the following steps in determining if it will allow a connection to a specified service. If all the steps fail, the connection request is rejected. However, if one of the steps succeeds, the following steps are not checked.

If the service is marked “*guest only = yes*” and the server is running with share-level security (“*security = share*”, steps 1 to 5 are skipped).

- 1 If the client has passed a username/password pair and that username/password pair is validated by the UNIX system’s password programs, the connection is made as that username. This includes the `\\server\service%username` method of passing a username.
- 2 If the client has previously registered a username with the system and now supplies a correct password for that username, the connection is allowed.
- 3 The client’s NetBIOS name and any previously used usernames are checked against the supplied password. If they match, the connection is allowed as the corresponding user.
- 4 If the client has previously validated a username/password pair with the server and the client has passed the validation token, that username is used.
- 5 If a “*user =*” field is given in the `smb.conf` file for the service and the client has supplied a password, and that password matches (according to the UNIX system’s password checking) with one of the usernames from the “*user =*” field, the connection is made as the username in the “*user =*” line. If one of the usernames in the “*user =*” list begins with a “*@*”, that name expands to a list of names in the group of the same name.
- 6 If the service is a guest service, a connection is made as the username given in the “*guest account =*” for the service, irrespective of the supplied password.

EXPLANATION OF EACH PARAMETER

abort shutdown script (G) — *This parameter only exists in the HEAD cvs branch* This a full path name to a script called by `smbd(8)` that should stop a shutdown procedure issued by the *shutdown script*.

This command will be run as user.

Default: *None*.

Example: **abort shutdown script = /sbin/shutdown -c**

add group script (G) — This is the full pathname to a script that will be run *AS ROOT* by `smbd(8)` when a new group is requested. It will expand any `%g` to the group name passed. This script is only useful for installations using the Windows NT domain administration tools. The script is free to create a group with an arbitrary name to circumvent unix group name restrictions. In that case the script must print the numeric gid of the created group on stdout.

add machine script (G) — This is the full pathname to a script that will be run by `smbd(8)` when a machine is added to it's domain using the administrator username and password method.

This option is only required when using sam back-ends tied to the Unix uid method of RID calculation such as `smbpasswd`. This option is only available in Samba 3.0.

Default: **add machine script = <empty string>**

Example: **add machine script = /usr/sbin/adduser -n -g machines -c Machine -d /dev/null -s /bin/false %u**

addprinter command (G) — With the introduction of MS-RPC based printing support for Windows NT/2000 clients in Samba 2.2, The MS Add Printer Wizard (APW) icon is now also available in the "Printers..." folder displayed a share listing. The APW allows for printers to be add remotely to a Samba or Windows NT/2000 print server.

For a Samba host this means that the printer must be physically added to the underlying printing system. The *add printer command* defines a script to be run which will perform the necessary operations for adding the printer to the print system and to add the appropriate service definition to the `smb.conf` file in order that it can be shared by `smbd(8)`.

The *addprinter command* is automatically invoked with the following parameter (in order):

- *printer name*
- *share name*
- *port name*
- *driver name*
- *location*
- *Windows 9x driver location*

All parameters are filled in from the `PRINTER_INFO_2` structure sent by the Windows NT/2000 client with one exception. The "Windows 9x driver location" parameter is

included for backwards compatibility only. The remaining fields in the structure are generated from answers to the APW questions.

Once the *addprinter command* has been executed, **smbd** will reparse the *smb.conf* to determine if the share defined by the APW exists. If the sharename is still invalid, then **smbd** will return an ACCESS_DENIED error to the client.

The "add printer command" program can output a single line of text, which Samba will set as the port the new printer is connected to. If this line isn't output, Samba won't reload its printer shares.

See also *deleteprinter command*, *printing*, *show add printer wizard*

Default: *none*

Example: **addprinter command = /usr/bin/addprinter**

add share command (G) — Samba 2.2.0 introduced the ability to dynamically add and delete shares via the Windows NT 4.0 Server Manager. The *add share command* is used to define an external program or script which will add a new service definition to *smb.conf*. In order to successfully execute the *add share command*, **smbd** requires that the administrator be connected using a root account (i.e. uid == 0).

When executed, **smbd** will automatically invoke the *add share command* with four parameters.

- *configFile* - the location of the global *smb.conf* file.
- *shareName* - the name of the new share.
- *pathName* - path to an ***existing*** directory on disk.
- *comment* - comment string to associate with the new share.

This parameter is only used for add file shares. To add printer shares, see the *addprinter command*.

See also *change share command*, *delete share command*.

Default: *none*

Example: **add share command = /usr/local/bin/addshare**

add user script (G) — This is the full pathname to a script that will be run *AS ROOT* by **smbd(8)** under special circumstances described below.

Normally, a Samba server requires that UNIX users are created for all users accessing files on this server. For sites that use Windows NT account databases as their primary user database creating these users and keeping the user list in sync with the Windows NT PDC is an onerous task. This option allows **smbd** to create the required UNIX users *ON DEMAND* when a user accesses the Samba server.

In order to use this option, **smbd(8)** must *NOT* be set to *security = share* and *add user script* must be set to a full pathname for a script that will create a UNIX user given one argument of *%u*, which expands into the UNIX user name to create.

When the Windows user attempts to access the Samba server, at login (session setup in the SMB protocol) time, `smbd(8)` contacts the *password server* and attempts to authenticate the given user with the given password. If the authentication succeeds then **smbd** attempts to find a UNIX user in the UNIX password database to map the Windows user into. If this lookup fails, and *add user script* is set then **smbd** will call the specified script *AS ROOT*, expanding any `%u` argument to be the user name to create.

If this script successfully creates the user then **smbd** will continue on as though the UNIX user already existed. In this way, UNIX users are dynamically created to match existing Windows NT accounts.

See also *security*, *password server*, *delete user script*.

Default: **add user script** = <empty string>

Example: **add user script** = `/usr/local/samba/bin/add_user %u`

add user to group script (G) — Full path to the script that will be called when a user is added to a group using the Windows NT domain administration tools. It will be run by `smbd(8)` *AS ROOT*. Any `%g` will be replaced with the group name and any `%u` will be replaced with the user name.

Default: **add user to group script** =

Example: **add user to group script** = `/usr/sbin/adduser %u %g`

admin users (S) — This is a list of users who will be granted administrative privileges on the share. This means that they will do all file operations as the super-user (root).

You should use this option very carefully, as any user in this list will be able to do anything they like on the share, irrespective of file permissions.

Default: *no admin users*

Example: **admin users** = `jason`

algorithmic rid base (G) — This determines how Samba will use its algorithmic mapping from uids/gid to the RIDs needed to construct NT Security Identifiers.

Setting this option to a larger value could be useful to sites transitioning from WinNT and Win2k, as existing user and group rids would otherwise clash with system users etc.

All UIDs and GIDs must be able to be resolved into SIDs for the correct operation of ACLs on the server. As such the algorithmic mapping can't be 'turned off', but pushing it 'out of the way' should resolve the issues. Users and groups can then be assigned 'low' RIDs in arbitrary-rid supporting backends.

Default: **algorithmic rid base** = `1000`

Example: **algorithmic rid base** = `100000`

allow hosts (S) — Synonym for *hosts allow*.

allow trusted domains (G) — This option only takes effect when the *security* option is set to **server**, **domain**, or **ads**. If it is set to **no**, then attempts to connect to a resource from a domain or workgroup other than the one which *smbd* is running in will fail, even if that domain is trusted by the remote server doing the authentication.

This is useful if you only want your Samba server to serve resources to users in the domain it is a member of. As an example, suppose that there are two domains DOMA and DOMB. DOMB is trusted by DOMA, which contains the Samba server. Under normal circumstances, a user with an account in DOMB can then access the resources of a UNIX account with the same account name on the Samba server even if they do not have an account in DOMA. This can make implementing a security boundary difficult.

Default: **allow trusted domains = yes**

announce as (G) — This specifies what type of server *nmbd*(8) will announce itself as, to a network neighborhood browse list. By default this is set to Windows NT. The valid options are : "NT Server" (which can also be written as "NT"), "NT Workstation", "Win95" or "WfW" meaning Windows NT Server, Windows NT Workstation, Windows 95 and Windows for Workgroups respectively. Do not change this parameter unless you have a specific need to stop Samba appearing as an NT server as this may prevent Samba servers from participating as browser servers correctly.

Default: **announce as = NT Server**

Example: **announce as = Win95**

announce version (G) — This specifies the major and minor version numbers that *nmbd* will use when announcing itself as a server. The default is 4.9. Do not change this parameter unless you have a specific need to set a Samba server to be a downlevel server.

Default: **announce version = 4.9**

Example: **announce version = 2.0**

auth methods (G) — This option allows the administrator to chose what authentication methods *smbd* will use when authenticating a user. This option defaults to sensible values based on *security*. This should be considered a developer option and used only in rare circumstances. In the majority (if not all) of production servers, the default setting should be adequate.

Each entry in the list attempts to authenticate the user in turn, until the user authenticates. In practice only one method will ever actually be able to complete the authentication.

Possible options include **guest** (anonymous access), **sam** (lookups in local list of accounts based on netbios name or domain name), **winbind** (relay authentication requests for remote users through winbindd), **ntdomain** (pre-winbindd method of authentication for remote domain users; deprecated in favour of winbind method), **trustdomain** (authenticate trusted users by contacting the remote DC directly from smbd; deprecated in favour of winbind method).

Default: **auth methods = <empty string>**

Example: **auth methods = guest sam winbind**

auto services (G) — This is a synonym for the *preload*.

available (S) — This parameter lets you "turn off" a service. If *available = no*, then *ALL* attempts to connect to the service will fail. Such failures are logged.

Default: **available = yes**

bind interfaces only (G) — This global parameter allows the Samba admin to limit what interfaces on a machine will serve SMB requests. It affects file service smbd(8) and name service nmbd(8) in a slightly different ways.

For name service it causes **nmbd** to bind to ports 137 and 138 on the interfaces listed in the *interfaces* parameter. **nmbd** also binds to the "all addresses" interface (0.0.0.0) on ports 137 and 138 for the purposes of reading broadcast messages. If this option is not set then **nmbd** will service name requests on all of these sockets. If *bind interfaces only* is set then **nmbd** will check the source address of any packets coming in on the broadcast sockets and discard any that don't match the broadcast addresses of the interfaces in the *interfaces* parameter list. As unicast packets are received on the other sockets it allows **nmbd** to refuse to serve names to machines that send packets that arrive through any interfaces not listed in the *interfaces* list. IP Source address spoofing does defeat this simple check, however, so it must not be used seriously as a security feature for **nmbd**.

For file service it causes smbd(8) to bind only to the interface list given in the *interfaces* parameter. This restricts the networks that **smbd** will serve to packets coming in those interfaces. Note that you should not use this parameter for machines that are serving PPP or other intermittent or non-broadcast network interfaces as it will not cope with non-permanent interfaces.

If *bind interfaces only* is set then unless the network address *127.0.0.1* is added to the *interfaces* parameter list smbpasswd(8) and swat(8) may not work as expected due to the reasons covered below.

To change a users SMB password, the **smbpasswd** by default connects to the *localhost* - *127.0.0.1* address as an SMB client to issue the password change request. If *bind interfaces only* is set then unless the network address *127.0.0.1* is added to the *interfaces* parameter list then **smbpasswd** will fail to connect in it's default mode. **smbpasswd** can be forced to use the primary IP interface of the local host by using

its `smbpasswd(8)` `-r remote machine` parameter, with `remote machine` set to the IP name of the primary interface of the local host.

The `swat` status page tries to connect with `smbd` and `nmbd` at the address `127.0.0.1` to determine if they are running. Not adding `127.0.0.1` will cause `smbd` and `nmbd` to always show "not running" even if they really are. This can prevent `swat` from starting/stopping/restarting `smbd` and `nmbd`.

Default: **bind interfaces only = no**

blocking locks (S) — This parameter controls the behavior of `smbd(8)` when given a request by a client to obtain a byte range lock on a region of an open file, and the request has a time limit associated with it.

If this parameter is set and the lock range requested cannot be immediately satisfied, samba will internally queue the lock request, and periodically attempt to obtain the lock until the timeout period expires.

If this parameter is set to `no`, then samba will behave as previous versions of Samba would and will fail the lock request immediately if the lock range cannot be obtained.

Default: **blocking locks = yes**

block size (S) — This parameter controls the behavior of `smbd(8)` when reporting disk free sizes. By default, this reports a disk block size of 1024 bytes.

Changing this parameter may have some effect on the efficiency of client writes, this is not yet confirmed. This parameter was added to allow advanced administrators to change it (usually to a higher value) and test the effect it has on client write performance without re-compiling the code. As this is an experimental option it may be removed in a future release.

Changing this option does not change the disk free reporting size, just the block size unit reported to the client.

browsable (S) — See the *browseable*.

browseable (S) — This controls whether this share is seen in the list of available shares in a net view and in the browse list.

Default: **browseable = yes**

browse list (G) — This controls whether `smbd(8)` will serve a browse list to a client doing a `NetServerEnum` call. Normally set to `yes`. You should never need to change this.

Default: **browse list = yes**

case sensitive (S) — See the discussion in the section NAME MANGLING.

Default: **case sensitive = no**

casesignames (S) — Synonym for case sensitive.

change notify timeout (G) — This SMB allows a client to tell a server to "watch" a particular directory for any changes and only reply to the SMB request when a change has occurred. Such constant scanning of a directory is expensive under UNIX, hence an `smbd(8)` daemon only performs such a scan on each requested directory once every *change notify timeout* seconds.

Default: **change notify timeout = 60**

Example: **change notify timeout = 300**

Would change the scan time to every 5 minutes.

change share command (G) — Samba 2.2.0 introduced the ability to dynamically add and delete shares via the Windows NT 4.0 Server Manager. The *change share command* is used to define an external program or script which will modify an existing service definition in `smb.conf`. In order to successfully execute the *change share command*, `smbd` requires that the administrator be connected using a root account (i.e. `uid == 0`).

When executed, `smbd` will automatically invoke the *change share command* with four parameters.

- *configFile* - the location of the global `smb.conf` file.
- *shareName* - the name of the new share.
- *pathName* - path to an ****existing**** directory on disk.
- *comment* - comment string to associate with the new share.

This parameter is only used modify existing file shares definitions. To modify printer shares, use the "Printers..." folder as seen when browsing the Samba host.

See also *add share command*, *delete share command*.

Default: *none*

Example: **change share command = /usr/local/bin/addshare**

client lanman auth (G) — This parameter determines whether or not `smbclient(8)` and other samba client tools will attempt to authenticate itself to servers using the weaker LANMAN password hash. If disabled, only server which support NT password hashes (e.g. Windows NT/2000, Samba, etc... but not Windows 95/98) will be able to be connected from the Samba client.

The LANMAN encrypted response is easily broken, due to it's case-insensitive nature, and the choice of algorithm. Clients without Windows 95/98 servers are advised to disable this option.

Disabling this option will also disable the **client plaintext auth** option

Likewise, if the **client ntlmv2 auth** parameter is enabled, then only NTLMv2 logins will be attempted. Not all servers support NTLMv2, and most will require special configuration to us it.

Default : **client lanman auth = yes**

client ntlmv2 auth (G) — This parameter determines whether or not smbclient(8) will attempt to authenticate itself to servers using the NTLMv2 encrypted password response.

If enabled, only an NTLMv2 and LMv2 response (both much more secure than earlier versions) will be sent. Many servers (including NT4 < SP4, Win9x and Samba 2.2) are not compatible with NTLMv2.

Similarly, if enabled, NTLMv1, **client lanman auth** and **client plaintext auth** authentication will be disabled. This also disables share-level authentication.

If disabled, an NTLM response (and possibly a LANMAN response) will be sent by the client, depending on the value of **client lanman auth**.

Note that some sites (particularly those following 'best practice' security policies) only allow NTLMv2 responses, and not the weaker LM or NTLM.

Default : **client ntlmv2 auth = no**

client use spnego (G) — This variable controls controls whether samba clients will try to use Simple and Protected NEGOCIation (as specified by rfc2478) with WindowsXP and Windows2000 servers to agree upon an authentication mechanism. SPNEGO client support for SMB Signing is currently broken, so you might want to turn this option off when operating with Windows 2003 domain controllers in particular.

Default: *client use spnego = yes*

comment (S) — This is a text field that is seen next to a share when a client does a queries the server, either via the network neighborhood or via **net view** to list what shares are available.

If you want to set the string that is displayed next to the machine name then see the *server string* parameter.

Default: *No comment string*

Example: **comment = Fred's Files**

config file (G) — This allows you to override the config file to use, instead of the default (usually **smb.conf**). There is a chicken and egg problem here as this option is set in the config file!

For this reason, if the name of the config file has changed when the parameters are loaded then it will reload them from the new config file.

This option takes the usual substitutions, which can be very useful.

If the config file doesn't exist then it won't be loaded (allowing you to special case the config files of just a few clients).

Example: **config file** = `/usr/local/samba/lib/smb.conf.%m`

copy (S) — This parameter allows you to "clone" service entries. The specified service is simply duplicated under the current service's name. Any parameters specified in the current section will override those in the section being copied.

This feature lets you set up a 'template' service and create similar services easily. Note that the service being copied must occur earlier in the configuration file than the service doing the copying.

Default: *no value*

Example: **copy** = **otherservice**

create mask (S) — A synonym for this parameter is *create mode*.

When a file is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is then bit-wise 'AND'ed with this parameter. This parameter may be thought of as a bit-wise MASK for the UNIX modes of a file. Any bit *not* set here will be removed from the modes set on a file when it is created.

The default value of this parameter removes the 'group' and 'other' write and execute bits from the UNIX modes.

Following this Samba will bit-wise 'OR' the UNIX mode created from this parameter with the value of the *force create mode* parameter which is set to 000 by default.

This parameter does not affect directory modes. See the parameter *directory mode* for details.

See also the *force create mode* parameter for forcing particular mode bits to be set on created files. See also the *directory mode* parameter for masking mode bits on created directories. See also the *inherit permissions* parameter.

Note that this parameter does not apply to permissions set by Windows NT/2000 ACL editors. If the administrator wishes to enforce a mask on access control lists also, they need to set the *security mask*.

Default: **create mask** = **0744**

Example: **create mask** = **0775**

create mode (S) — This is a synonym for *create mask*.

csc policy (S) — This stands for *client-side caching policy*, and specifies how clients capable of offline caching will cache the files in the share. The valid values are: manual, documents, programs, disable.

These values correspond to those used on Windows servers.

For example, shares containing roaming profiles can have offline caching disabled using **csc policy = disable**.

Default: **csc policy = manual**

Example: **csc policy = programs**

deadtime (G) — The value of the parameter (a decimal integer) represents the number of minutes of inactivity before a connection is considered dead, and it is disconnected. The deadtime only takes effect if the number of open files is zero.

This is useful to stop a server's resources being exhausted by a large number of inactive connections.

Most clients have an auto-reconnect feature when a connection is broken so in most cases this parameter should be transparent to users.

Using this parameter with a timeout of a few minutes is recommended for most systems.

A deadtime of zero indicates that no auto-disconnection should be performed.

Default: **deadtime = 0**

Example: **deadtime = 15**

debug hires timestamp (G) — Sometimes the timestamps in the log messages are needed with a resolution of higher than seconds, this boolean parameter adds microsecond resolution to the timestamp message header when turned on.

Note that the parameter *debug timestamp* must be on for this to have an effect.

Default: **debug hires timestamp = no**

debuglevel (G) — Synonym for *log level*.

debug pid (G) — When using only one log file for more than one forked `smbd(8)`-process there may be hard to follow which process outputs which message. This boolean parameter is added the process-id to the timestamp message headers in the logfile when turned on.

Note that the parameter *debug timestamp* must be on for this to have an effect.

Default: **debug pid = no**

debug timestamp (G) — Samba debug log messages are timestamped by default. If you are running at a high *debug level* these timestamps can be distracting. This boolean parameter allows timestamping to be turned off.

Default: **debug timestamp = yes**

debug uid (G) — Samba is sometimes run as root and sometime run as the connected user, this boolean parameter inserts the current euid, egid, uid and gid to the timestamp message headers in the log file if turned on.

Note that the parameter *debug timestamp* must be on for this to have an effect.

Default: **debug uid = no**

default (G) — A synonym for *default service*.

default case (S) — See the section on NAME MANGLING. Also note the *short pre-serve case* parameter.

Default: **default case = lower**

default devmode (S) — This parameter is only applicable to printable services. When `smbd` is serving Printer Drivers to Windows NT/2k/XP clients, each printer on the Samba server has a Device Mode which defines things such as paper size and orientation and duplex settings. The device mode can only correctly be generated by the printer driver itself (which can only be executed on a Win32 platform). Because `smbd` is unable to execute the driver code to generate the device mode, the default behavior is to set this field to NULL.

Most problems with serving printer drivers to Windows NT/2k/XP clients can be traced to a problem with the generated device mode. Certain drivers will do things such as crashing the client's Explorer.exe with a NULL devmode. However, other printer drivers can cause the client's spooler service (`spoolsv.exe`) to die if the devmode was not created by the driver itself (i.e. `smbd` generates a default devmode).

This parameter should be used with care and tested with the printer driver in question. It is better to leave the device mode to NULL and let the Windows client set the correct values. Because drivers do not do this all the time, setting **default devmode = yes** will instruct `smbd` to generate a default one.

For more information on Windows NT/2k printing and Device Modes, see the MSDN documentation³.

Default: **default devmode = no**

default service (G) — This parameter specifies the name of a service which will be connected to if the service actually requested cannot be found. Note that the square brackets are *NOT* given in the parameter value (see example below).

There is no default value for this parameter. If this parameter is not given, attempting to connect to a nonexistent service results in an error.

Typically the default service would be a *guest ok, read-only* service.

³<http://msdn.microsoft.com/>

Also note that the apparent service name will be changed to equal that of the requested service, this is very useful as it allows you to use macros like `%S` to make a wildcard service.

Note also that any `"_"` characters in the name of the service used in the default service will get mapped to a `"/"`. This allows for interesting things.

Example:

```
[global]
    default service = pub

[pub]
    path = /%S
```

delete group script (G) — This is the full pathname to a script that will be run *AS ROOT* `smbd(8)` when a group is requested to be deleted. It will expand any `%g` to the group name passed. This script is only useful for installations using the Windows NT domain administration tools.

deleteprinter command (G) — With the introduction of MS-RPC based printer support for Windows NT/2000 clients in Samba 2.2, it is now possible to delete printer at run time by issuing the `DeletePrinter()` RPC call.

For a Samba host this means that the printer must be physically deleted from underlying printing system. The *deleteprinter command* defines a script to be run which will perform the necessary operations for removing the printer from the print system and from `smb.conf`.

The *deleteprinter command* is automatically called with only a single parameter: *"printer name"*.

Once the *deleteprinter command* has been executed, `smbd` will reparse the `smb.conf` to associated printer no longer exists. If the sharename is still valid, then `smbd` will return an `ACCESS_DENIED` error to the client.

See also *addprinter command*, *printing*, *show add printer wizard*

Default: *none*

Example: `deleteprinter command = /usr/bin/removeprinter`

delete readonly (S) — This parameter allows readonly files to be deleted. This is not normal DOS semantics, but is allowed by UNIX.

This option may be useful for running applications such as `rcs`, where UNIX file ownership prevents changing file permissions, and DOS semantics prevent deletion of a read only file.

Default: `delete readonly = no`

delete share command (G) — Samba 2.2.0 introduced the ability to dynamically add and delete shares via the Windows NT 4.0 Server Manager. The *delete share command* is used to define an external program or script which will remove an existing service definition from *smb.conf*. In order to successfully execute the *delete share command*, *smbd* requires that the administrator be connected using a root account (i.e. `uid == 0`).

When executed, *smbd* will automatically invoke the *delete share command* with two parameters.

- *configFile* - the location of the global *smb.conf* file.
- *shareName* - the name of the existing service.

This parameter is only used to remove file shares. To delete printer shares, see the *deleteprinter command*.

See also *add share command*, *change share command*.

Default: *none*

Example: **delete share command** = `/usr/local/bin/delshare`

delete user from group script (G) — Full path to the script that will be called when a user is removed from a group using the Windows NT domain administration tools. It will be run by *smbd*(8) *AS ROOT*. Any *%g* will be replaced with the group name and any *%u* will be replaced with the user name.

Default: **delete user from group script** =

Example: **delete user from group script** = `/usr/sbin/deluser %u %g`

delete user script (G) — This is the full pathname to a script that will be run by *smbd*(8) when managing users with remote RPC (NT) tools.

This script is called when a remote client removes a user from the server, normally using 'User Manager for Domains' or **rpcclient**.

This script should delete the given UNIX username.

Default: **delete user script** = `<empty string>`

Example: **delete user script** = `/usr/local/samba/bin/del_user %u`

delete veto files (S) — This option is used when Samba is attempting to delete a directory that contains one or more vetoed directories (see the *veto files* option). If this option is set to **no** (the default) then if a vetoed directory contains any non-vetoed files or directories then the directory delete will fail. This is usually what you want.

If this option is set to **yes**, then Samba will attempt to recursively delete any files and directories within the vetoed directory. This can be useful for integration with file

serving systems such as NetAtalk which create meta-files within directories you might normally veto DOS/Windows users from seeing (e.g. `.AppleDouble`)

Setting **delete veto files = yes** allows these directories to be transparently deleted when the parent directory is deleted (so long as the user has permissions to do so).

See also the *veto files* parameter.

Default: **delete veto files = no**

deny hosts (S) — Synonym for *hosts deny*.

dfree command (G) — The *dfree command* setting should only be used on systems where a problem occurs with the internal disk space calculations. This has been known to happen with Ultrix, but may occur with other operating systems. The symptom that was seen was an error of "Abort Retry Ignore" at the end of each directory listing.

This setting allows the replacement of the internal routines to calculate the total disk space and amount available with an external routine. The example below gives a possible script that might fulfill this function.

The external program will be passed a single parameter indicating a directory in the filesystem being queried. This will typically consist of the string `./`. The script should return two integers in ASCII. The first should be the total disk space in blocks, and the second should be the number of available blocks. An optional third return value can give the block size in bytes. The default blocksize is 1024 bytes.

Note: Your script should *NOT* be `setuid` or `setgid` and should be owned by (and writeable only by) root!

Default: *By default internal routines for determining the disk capacity and remaining space will be used.*

Example: **dfree command = /usr/local/samba/bin/dfree**

Where the script `dfree` (which must be made executable) could be:

```
#!/bin/sh
df $1 | tail -1 | awk '{print $2" "$4}'
```

or perhaps (on Sys V based systems):

```
#!/bin/sh
/usr/bin/df -k $1 | tail -1 | awk '{print $3" "$5}'
```

Note that you may have to replace the command names with full path names on some systems.

directory (S) — Synonym for *path*.

directory mask (S) — This parameter is the octal modes which are used when converting DOS modes to UNIX modes when creating UNIX directories.

When a directory is created, the necessary permissions are calculated according to the mapping from DOS modes to UNIX permissions, and the resulting UNIX mode is then bit-wise 'AND'ed with this parameter. This parameter may be thought of as a bit-wise MASK for the UNIX modes of a directory. Any bit *not* set here will be removed from the modes set on a directory when it is created.

The default value of this parameter removes the 'group' and 'other' write bits from the UNIX mode, allowing only the user who owns the directory to modify it.

Following this Samba will bit-wise 'OR' the UNIX mode created from this parameter with the value of the *force directory mode* parameter. This parameter is set to 000 by default (i.e. no extra mode bits are added).

Note that this parameter does not apply to permissions set by Windows NT/2000 ACL editors. If the administrator wishes to enforce a mask on access control lists also, they need to set the *directory security mask*.

See the *force directory mode* parameter to cause particular mode bits to always be set on created directories.

See also the *create mode* parameter for masking mode bits on created files, and the *directory security mask* parameter.

Also refer to the *inherit permissions* parameter.

Default: **directory mask = 0755**

Example: **directory mask = 0775**

directory mode (S) — Synonym for *directory mask*

directory security mask (S) — This parameter controls what UNIX permission bits can be modified when a Windows NT client is manipulating the UNIX permission on a directory using the native NT security dialog box.

This parameter is applied as a mask (AND'ed with) to the changed permission bits, thus preventing any bits not in this mask from being modified. Essentially, zero bits in this mask may be treated as a set of bits the user is not allowed to change.

If not set explicitly this parameter is set to 0777 meaning a user is allowed to modify all the user/group/world permissions on a directory.

Note that users who can access the Samba server through other means can easily bypass this restriction, so it is primarily useful for standalone "appliance" systems. Administrators of most normal systems will probably want to leave it as the default of 0777.

See also the *force directory security mode*, *security mask*, *force security mode* parameters.

Default: **directory security mask = 0777**

Example: **directory security mask = 0700**

disable netbios (G) — Enabling this parameter will disable netbios support in Samba. Netbios is the only available form of browsing in all windows versions except for 2000 and XP.

NOTE



Note that clients that only support netbios won't be able to see your samba server when netbios support is disabled.

Default: **disable netbios = no**

Example: **disable netbios = yes**

disable spoolss (G) — Enabling this parameter will disable Samba's support for the SPOOLSS set of MS-RPC's and will yield identical behavior as Samba 2.0.x. Windows NT/2000 clients will downgrade to using Lanman style printing commands. Windows 9x/ME will be unaffected by the parameter. However, this will also disable the ability to upload printer drivers to a Samba server via the Windows NT Add Printer Wizard or by using the NT printer properties dialog window. It will also disable the capability of Windows NT/2000 clients to download print drivers from the Samba host upon demand. *Be very careful about enabling this parameter.*

See also use client driver

Default : **disable spoolss = no**

display charset (G) — Specifies the charset that samba will use to print messages to stdout and stderr and SWAT will use. Should generally be the same as the **unix charset**.

Default: **display charset = ASCII**

Example: **display charset = UTF8**

dns proxy (G) — Specifies that nmbd(8) when acting as a WINS server and finding that a NetBIOS name has not been registered, should treat the NetBIOS name word-for-word as a DNS name and do a lookup with the DNS server for that name on behalf of the name-querying client.

Note that the maximum length for a NetBIOS name is 15 characters, so the DNS name (or DNS alias) can likewise only be 15 characters, maximum.

nmbd spawns a second copy of itself to do the DNS name lookup requests, as doing a name lookup is a blocking action.

See also the parameter *wins support*.

Default: **dns proxy = yes**

domain logons (G) — If set to **yes**, the Samba server will serve Windows 95/98 Domain logons for the *workgroup* it is in. Samba 2.2 has limited capability to act as a domain controller for Windows NT 4 Domains. For more details on setting up this feature see the Samba-PDC-HOWTO included in the Samba documentation.

Default: **domain logons = no**

domain master (G) — Tell `smbd(8)` to enable WAN-wide browse list collation. Setting this option causes **nmbd** to claim a special domain specific NetBIOS name that identifies it as a domain master browser for its given *workgroup*. Local master browsers in the same *workgroup* on broadcast-isolated subnets will give this **nmbd** their local browse lists, and then ask `smbd(8)` for a complete copy of the browse list for the whole wide area network. Browser clients will then contact their local master browser, and will receive the domain-wide browse list, instead of just the list for their broadcast-isolated subnet.

Note that Windows NT Primary Domain Controllers expect to be able to claim this *workgroup* specific special NetBIOS name that identifies them as domain master browsers for that *workgroup* by default (i.e. there is no way to prevent a Windows NT PDC from attempting to do this). This means that if this parameter is set and **nmbd** claims the special name for a *workgroup* before a Windows NT PDC is able to do so then cross subnet browsing will behave strangely and may fail.

If **domain logons = yes**, then the default behavior is to enable the *domain master* parameter. If *domain logons* is not enabled (the default setting), then neither will *domain master* be enabled by default.

Default: **domain master = auto**

dont descend (S) — There are certain directories on some systems (e.g., the `/proc` tree under Linux) that are either not of interest to clients or are infinitely deep (recursive). This parameter allows you to specify a comma-delimited list of directories that the server should always show as empty.

Note that Samba can be very fussy about the exact format of the "dont descend" entries. For example you may need `./proc` instead of just `/proc`. Experimentation is the best policy :-)

Default: *none (i.e., all directories are OK to descend)*

Example: **dont descend = /proc,/dev**

dos charset (G) — DOS SMB clients assume the server has the same charset as they do. This option specifies which charset Samba should talk to DOS clients.

The default depends on which charsets you have installed. Samba tries to use charset 850 but falls back to ASCII in case it is not available. Run `testparm(1)` to check the default on your system.

dos filemode (S) — The default behavior in Samba is to provide UNIX-like behavior where only the owner of a file/directory is able to change the permissions on it. However, this behavior is often confusing to DOS/Windows users. Enabling this parameter allows a user who has write access to the file (by whatever means) to modify the permissions on it. Note that a user belonging to the group owning the file will not be allowed to change permissions if the group is only granted read access. Ownership of the file/directory is not changed, only the permissions are modified.

Default: **dos filemode = no**

dos filetime resolution (S) — Under the DOS and Windows FAT filesystem, the finest granularity on time resolution is two seconds. Setting this parameter for a share causes Samba to round the reported time down to the nearest two second boundary when a query call that requires one second resolution is made to `smbd(8)`.

This option is mainly used as a compatibility option for Visual C++ when used against Samba shares. If oplocks are enabled on a share, Visual C++ uses two different time reading calls to check if a file has changed since it was last read. One of these calls uses a one-second granularity, the other uses a two second granularity. As the two second call rounds any odd second down, then if the file has a timestamp of an odd number of seconds then the two timestamps will not match and Visual C++ will keep reporting the file has changed. Setting this option causes the two timestamps to match, and Visual C++ is happy.

Default: **dos filetime resolution = no**

dos filetimes (S) — Under DOS and Windows, if a user can write to a file they can change the timestamp on it. Under POSIX semantics, only the owner of the file or root may change the timestamp. By default, Samba runs with POSIX semantics and refuses to change the timestamp on a file if the user `smbd` is acting on behalf of is not the file owner. Setting this option to **yes** allows DOS semantics and `smbd(8)` will change the file timestamp as DOS requires.

Default: **dos filetimes = no**

encrypt passwords (G) — This boolean controls whether encrypted passwords will be negotiated with the client. Note that Windows NT 4.0 SP3 and above and also Windows 98 will by default expect encrypted passwords unless a registry entry is changed. To use encrypted passwords in Samba see the chapter "User Database" in the Samba HOWTO Collection.

In order for encrypted passwords to work correctly `smbd(8)` must either have access

to a local `smbpasswd(5)` file (see the `smbpasswd(8)` program for information on how to set up and maintain this file), or set the `security = [server|domain|ads]` parameter which causes `smbd` to authenticate against another server.

Default: **encrypt passwords = yes**

enhanced browsing (G) — This option enables a couple of enhancements to cross-subnet browse propagation that have been added in Samba but which are not standard in Microsoft implementations.

The first enhancement to browse propagation consists of a regular wildcard query to a Samba WINS server for all Domain Master Browsers, followed by a browse synchronization with each of the returned DMBs. The second enhancement consists of a regular randomised browse synchronization with all currently known DMBs.

You may wish to disable this option if you have a problem with empty workgroups not disappearing from browse lists. Due to the restrictions of the browse protocols these enhancements can cause a empty workgroup to stay around forever which can be annoying.

In general you should leave this option enabled as it makes cross-subnet browse propagation much more reliable.

Default: **enhanced browsing = yes**

enumports command (G) — The concept of a "port" is fairly foreign to UNIX hosts. Under Windows NT/2000 print servers, a port is associated with a port monitor and generally takes the form of a local port (i.e. LPT1:, COM1:, FILE:) or a remote port (i.e. LPD Port Monitor, etc...). By default, Samba has only one port defined—"Samba Printer Port". Under Windows NT/2000, all printers must have a valid port name. If you wish to have a list of ports displayed (`smbd` does not use a port name for anything) other than the default "Samba Printer Port", you can define *enumports command* to point to a program which should generate a list of ports, one per line, to standard output. This listing will then be used in response to the level 1 and 2 EnumPorts() RPC.

Default: *no enumports command*

Example: **enumports command = /usr/bin/listports**

exec (S) — This is a synonym for *preexec*.

fake directory create times (S) — NTFS and Windows VFAT file systems keep a create time for all files and directories. This is not the same as the `ctime` - status change time - that Unix keeps, so Samba by default reports the earliest of the various times Unix does keep. Setting this parameter for a share causes Samba to always report midnight 1-1-1980 as the create time for directories.

This option is mainly used as a compatibility option for Visual C++ when used against Samba shares. Visual C++ generated makefiles have the object directory as a depen-

dency for each object file, and a make rule to create the directory. Also, when NMAKE compares timestamps it uses the creation time when examining a directory. Thus the object directory will be created if it does not exist, but once it does exist it will always have an earlier timestamp than the object files it contains.

However, Unix time semantics mean that the create time reported by Samba will be updated whenever a file is created or or deleted in the directory. NMAKE finds all object files in the object directory. The timestamp of the last one built is then compared to the timestamp of the object directory. If the directory's timestamp is newer, then all object files will be rebuilt. Enabling this option ensures directories always predate their contents and an NMAKE build will proceed as expected.

Default: **fake directory create times = no**

fake oplocks (S) — Oplocks are the way that SMB clients get permission from a server to locally cache file operations. If a server grants an oplock (opportunistic lock) then the client is free to assume that it is the only one accessing the file and it will aggressively cache file data. With some oplock types the client may even cache file open/close operations. This can give enormous performance benefits.

When you set **fake oplocks = yes**, `smbd(8)` will always grant oplock requests no matter how many clients are using the file.

It is generally much better to use the real *oplocks* support rather than this parameter.

If you enable this option on all read-only shares or shares that you know will only be accessed from one client at a time such as physically read-only media like CDROMs, you will see a big performance improvement on many operations. If you enable this option on shares where multiple clients may be accessing the files read-write at the same time you can get data corruption. Use this option carefully!

Default: **fake oplocks = no**

follow symlinks (S) — This parameter allows the Samba administrator to stop `smbd(8)` from following symbolic links in a particular share. Setting this parameter to **no** prevents any file or directory that is a symbolic link from being followed (the user will get an error). This option is very useful to stop users from adding a symbolic link to `/etc/passwd` in their home directory for instance. However it will slow filename lookups down slightly.

This option is enabled (i.e. **smbd** will follow symbolic links) by default.

Default: **follow symlinks = yes**

force create mode (S) — This parameter specifies a set of UNIX mode bit permissions that will *always* be set on a file created by Samba. This is done by bitwise 'OR'ing these bits onto the mode bits of a file that is being created or having its permissions changed. The default for this parameter is (in octal) 000. The modes in this parameter are bitwise 'OR'ed onto the file mode after the mask set in the *create mask* parameter is applied.

See also the parameter *create mask* for details on masking mode bits on files.

See also the *inherit permissions* parameter.

Default: **force create mode = 000**

Example: **force create mode = 0755**

would force all created files to have read and execute permissions set for 'group' and 'other' as well as the read/write/execute bits set for the 'user'.

force directory mode (S) — This parameter specifies a set of UNIX mode bit permissions that will *always* be set on a directory created by Samba. This is done by bitwise 'OR'ing these bits onto the mode bits of a directory that is being created. The default for this parameter is (in octal) 0000 which will not add any extra permission bits to a created directory. This operation is done after the mode mask in the parameter *directory mask* is applied.

See also the parameter *directory mask* for details on masking mode bits on created directories.

See also the *inherit permissions* parameter.

Default: **force directory mode = 000**

Example: **force directory mode = 0755**

would force all created directories to have read and execute permissions set for 'group' and 'other' as well as the read/write/execute bits set for the 'user'.

force directory security mode (S) — This parameter controls what UNIX permission bits can be modified when a Windows NT client is manipulating the UNIX permission on a directory using the native NT security dialog box.

This parameter is applied as a mask (OR'ed with) to the changed permission bits, thus forcing any bits in this mask that the user may have modified to be on. Essentially, one bits in this mask may be treated as a set of bits that, when modifying security on a directory, the user has always set to be 'on'.

If not set explicitly this parameter is 000, which allows a user to modify all the user/group/world permissions on a directory without restrictions.

Note that users who can access the Samba server through other means can easily bypass this restriction, so it is primarily useful for standalone "appliance" systems. Administrators of most normal systems will probably want to leave it set as 0000.

See also the *directory security mask*, *security mask*, *force security mode* parameters.

Default: **force directory security mode = 0**

Example: **force directory security mode = 700**

force group (S) — This specifies a UNIX group name that will be assigned as the default primary group for all users connecting to this service. This is useful for sharing files by ensuring that all access to files on service will use the named group for their permissions checking. Thus, by assigning permissions for this group to the files and directories within this service the Samba administrator can restrict or allow sharing of these files.

In Samba 2.0.5 and above this parameter has extended functionality in the following way. If the group name listed here has a '+' character prepended to it then the current user accessing the share only has the primary group default assigned to this group if they are already assigned as a member of that group. This allows an administrator to decide that only users who are already in a particular group will create files with group ownership set to that group. This gives a finer granularity of ownership assignment. For example, the setting **force group = +sys** means that only users who are already in group sys will have their default primary group assigned to sys when accessing this Samba share. All other users will retain their ordinary primary group.

If the *force user* parameter is also set the group specified in *force group* will override the primary group set in *force user*.

See also *force user*.

Default: *no forced group*

Example: **force group = agroup**

force security mode (S) — This parameter controls what UNIX permission bits can be modified when a Windows NT client is manipulating the UNIX permission on a file using the native NT security dialog box.

This parameter is applied as a mask (OR'ed with) to the changed permission bits, thus forcing any bits in this mask that the user may have modified to be on. Essentially, one bits in this mask may be treated as a set of bits that, when modifying security on a file, the user has always set to be 'on'.

If not set explicitly this parameter is set to 0, and allows a user to modify all the user/group/world permissions on a file, with no restrictions.

Note that users who can access the Samba server through other means can easily bypass this restriction, so it is primarily useful for standalone "appliance" systems. Administrators of most normal systems will probably want to leave this set to 0000.

See also the *force directory security mode*, *directory security mask*, *security mask* parameters.

Default: **force security mode = 0**

Example: **force security mode = 700**

force user (S) — This specifies a UNIX user name that will be assigned as the default user for all users connecting to this service. This is useful for sharing files. You should also use it carefully as using it incorrectly can cause security problems.

This user name only gets used once a connection is established. Thus clients still need to connect as a valid user and supply a valid password. Once connected, all file operations will be performed as the "forced user", no matter what username the client connected as. This can be very useful.

In Samba 2.0.5 and above this parameter also causes the primary group of the forced user to be used as the primary group for all file activity. Prior to 2.0.5 the primary group was left as the primary group of the connecting user (this was a bug).

See also *force group*

Default: *no forced user*

Example: **force user = auser**

fstype (S) — This parameter allows the administrator to configure the string that specifies the type of filesystem a share is using that is reported by `smbd(8)` when a client queries the filesystem type for a share. The default type is `NTFS` for compatibility with Windows NT but this can be changed to other strings such as `Samba` or `FAT` if required.

Default: **fstype = NTFS**

Example: **fstype = Samba**

get quota command (G) — The **get quota command** should only be used whenever there is no operating system API available from the OS that samba can use.

This option is only available with `./configure --with-sys-quotas`. Or on linux when `./configure --with-quotas` was used and a working quota api was found in the system.

This parameter should specify the path to a script that queries the quota information for the specified user/group for the partition that the specified directory is on.

Such a script should take 3 arguments:

- directory
- type of query
- uid of user or gid of group

The type of query can be one of :

- 1 - user quotas
- 2 - user default quotas (uid = -1)
- 3 - group quotas
- 4 - group default quotas (gid = -1)

This script should print one line as output with spaces between the arguments. The arguments are:

- Arg 1 - quota flags (0 = no quotas, 1 = quotas enabled, 2 = quotas enabled and enforced)
- Arg 2 - number of currently used blocks
- Arg 3 - the softlimit number of blocks
- Arg 4 - the hardlimit number of blocks
- Arg 5 - currently used number of inodes
- Arg 6 - the softlimit number of inodes
- Arg 7 - the hardlimit number of inodes
- Arg 8(optional) - the number of bytes in a block(default is 1024)

See also the *set quota command* parameter.

Default: **get quota command =**

Example: **get quota command = /usr/local/sbin/query_quota**

getwd cache (G) — This is a tuning option. When this is enabled a caching algorithm will be used to reduce the time taken for `getwd()` calls. This can have a significant impact on performance, especially when the *wide links* parameter is set to **no**.

Default: **getwd cache = yes**

group (S) — Synonym for *force group*.

guest account (G,S) — This is a username which will be used for access to services which are specified as *guest ok* (see below). Whatever privileges this user has will be available to any client connecting to the guest service. Typically this user will exist in the password file, but will not have a valid login. The user account "ftp" is often a good choice for this parameter. If a username is specified in a given service, the specified username overrides this one.

On some systems the default guest account "nobody" may not be able to print. Use another account in this case. You should test this by trying to log in as your guest user (perhaps by using the **su** - command) and trying to print using the system print command such as **lpr(1)** or **lp(1)**.

This parameter does not accept % macros, because many parts of the system require this value to be constant for correct operation.

Default: *specified at compile time, usually "nobody"*

Example: **guest account = ftp**

guest ok (S) — If this parameter is **yes** for a service, then no password is required to connect to the service. Privileges will be those of the *guest account*.

This parameter nullifies the benefits of setting *restrict anonymous = 2*

See the section below on *security* for more information about this option.

Default: **guest ok = no**

guest only (S) — If this parameter is **yes** for a service, then only guest connections to the service are permitted. This parameter will have no effect if *guest ok* is not set for the service.

See the section below on *security* for more information about this option.

Default: **guest only = no**

hide dot files (S) — This is a boolean parameter that controls whether files starting with a dot appear as hidden files.

Default: **hide dot files = yes**

hide files (S) — This is a list of files or directories that are not visible but are accessible. The DOS 'hidden' attribute is applied to any files or directories that match.

Each entry in the list must be separated by a '/', which allows spaces to be included in the entry. '*' and '?' can be used to specify multiple files or directories as in DOS wildcards.

Each entry must be a Unix path, not a DOS path and must not include the Unix directory separator '/'.

Note that the case sensitivity option is applicable in hiding files.

Setting this parameter will affect the performance of Samba, as it will be forced to check all files and directories for a match as they are scanned.

See also *hide dot files*, *veto files* and *case sensitive*.

Default: *no file are hidden*

Example: **hide files = */DesktopFolderDB/TrashFor%m/resource.frk/**

The above example is based on files that the Macintosh SMB client (DAVE) available from Thursby⁴ creates for internal use, and also still hides all files beginning with a dot.

hide local users (G) — This parameter toggles the hiding of local UNIX users (root, wheel, floppy, etc) from remote clients.

Default: **hide local users = no**

hide special files (S) — This parameter prevents clients from seeing special files such as sockets, devices and fifo's in directory listings.

Default: **hide special files = no**

⁴<http://www.thursby.com>

hide unreadable (S) — This parameter prevents clients from seeing the existence of files that cannot be read. Defaults to off.

Default: **hide unreadable = no**

hide unwriteable files (S) — This parameter prevents clients from seeing the existence of files that cannot be written to. Defaults to off. Note that unwriteable directories are shown as usual.

Default: **hide unwriteable = no**

homedir map (G) — If *nis homedir* is **yes**, and `smbd(8)` is also acting as a Win95/98 *logon server* then this parameter specifies the NIS (or YP) map from which the server for the user's home directory should be extracted. At present, only the Sun auto.home map format is understood. The form of the map is:

username server:/some/file/system

and the program will extract the servername from before the first ':'. There should probably be a better parsing system that copes with different map formats and also Amd (another automounter) maps.

NOTE



A working NIS client is required on the system for this option to work.

See also *nis homedir*, *domain logons*.

Default: **homedir map = <empty string>**

Example: **homedir map = amd.homedir**

host msdfs (G) — If set to **yes**, Samba will act as a Dfs server, and allow Dfs-aware clients to browse Dfs trees hosted on the server.

See also the *msdfs root* share level parameter. For more information on setting up a Dfs tree on Samba, refer to Chapter 16, *Hosting a Microsoft Distributed File System Tree*.

Default: **host msdfs = no**

hostname lookups (G) — Specifies whether samba should use (expensive) hostname lookups or use the ip addresses instead. An example place where hostname lookups are currently used is when checking the **hosts deny** and **hosts allow**.

Default: **hostname lookups = yes**

Example: **hostname lookups = no**

hosts allow (S) — A synonym for this parameter is *allow hosts*.

This parameter is a comma, space, or tab delimited set of hosts which are permitted to access a service.

If specified in the [global] section then it will apply to all services, regardless of whether the individual service has a different setting.

You can specify the hosts by name or IP number. For example, you could restrict access to only the hosts on a Class C subnet with something like **allow hosts = 150.203.5.** The full syntax of the list is described in the man page `hosts_access(5)`. Note that this man page may not be present on your system, so a brief description will be given here also.

Note that the localhost address 127.0.0.1 will always be allowed access unless specifically denied by a *hosts deny* option.

You can also specify hosts by network/netmask pairs and by netgroup names if your system supports netgroups. The *EXCEPT* keyword can also be used to limit a wildcard list. The following examples may provide some help:

Example 1: allow all IPs in 150.203.*.*; except one

hosts allow = 150.203. EXCEPT 150.203.6.66

Example 2: allow hosts that match the given network/netmask

hosts allow = 150.203.15.0/255.255.255.0

Example 3: allow a couple of hosts

hosts allow = lapland, arvidsjaur

Example 4: allow only hosts in NIS netgroup "foonet", but deny access from one particular host

hosts allow = @foonet

hosts deny = pirate

NOTE



Note that access still requires suitable user-level passwords.

See `testparm(1)` for a way of testing your host access to see if it does what you expect.

Default: *none (i.e., all hosts permitted access)*

Example: **allow hosts = 150.203.5. myhost.mynet.edu.au**

hosts deny (S) — The opposite of *hosts allow* - hosts listed here are *NOT* permitted access to services unless the specific services have their own lists to override this one. Where the lists conflict, the *allow* list takes precedence.

Default: *none (i.e., no hosts specifically excluded)*

Example: **hosts deny = 150.203.4. badhost.mynet.edu.au**

hosts equiv (G) — If this global parameter is a non-null string, it specifies the name of a file to read for the names of hosts and users who will be allowed access without specifying a password.

This is not be confused with *hosts allow* which is about hosts access to services and is more useful for guest services. *hosts equiv* may be useful for NT clients which will not supply passwords to Samba.

NOTE



The use of *hosts equiv* can be a major security hole. This is because you are trusting the PC to supply the correct username. It is very easy to get a PC to supply a false username. I recommend that the *hosts equiv* option be only used if you really know what you are doing, or perhaps on a home network where you trust your spouse and kids. And only if you *really* trust them :-).

Default: *no host equivalences*

Example: **hosts equiv = /etc/hosts.equiv**

idmap backend (G) — The purpose of the *idmap backend* parameter is to allow *idmap* to NOT use the local *idmap tdb* file to obtain SID to UID / GID mappings, but instead to obtain them from a common LDAP backend. This way all domain members and controllers will have the same UID and GID to SID mappings. This avoids the risk of UID / GID inconsistencies across UNIX / Linux systems that are sharing information over protocols other than SMB/CIFS (ie: NFS).

Default: **idmap backend = <empty string>**

Example: **idmap backend = ldap:ldap://ldapslave.example.com**

idmap gid (G) — The *idmap gid* parameter specifies the range of group ids that are allocated for the purpose of mapping UNIX groups to NT group SIDs. This range of group ids should have no existing local or NIS groups within it as strange conflicts can occur otherwise.

The availability of an idmap gid range is essential for correct operation of all group mapping.

Default: **idmap gid** = <empty string>

Example: **idmap gid** = 10000-20000

idmap uid (G) — The idmap uid parameter specifies the range of user ids that are allocated for use in mapping UNIX users to NT user SIDs. This range of ids should have no existing local or NIS users within it as strange conflicts can occur otherwise.

Default: **idmap uid** = <empty string>

Example: **idmap uid** = 10000-20000

include (G) — This allows you to include one config file inside another. The file is included literally, as though typed in place.

It takes the standard substitutions, except *%u*, *%P* and *%S*.

Default: *no file included*

Example: **include** = /usr/local/samba/lib/admin_smb.conf

inherit acls (S) — This parameter can be used to ensure that if default acls exist on parent directories, they are always honored when creating a subdirectory. The default behavior is to use the mode specified when creating the directory. Enabling this option sets the mode to 0777, thus guaranteeing that default directory acls are propagated.

Default: **inherit acls** = no

inherit permissions (S) — The permissions on new files and directories are normally governed by *create mask*, *directory mask*, *force create mode* and *force directory mode* but the boolean inherit permissions parameter overrides this.

New directories inherit the mode of the parent directory, including bits such as setgid.

New files inherit their read/write bits from the parent directory. Their execute bits continue to be determined by *map archive*, *map hidden* and *map system* as usual.

Note that the setuid bit is *never* set via inheritance (the code explicitly prohibits this).

This can be particularly useful on large systems with many users, perhaps several thousand, to allow a single [homes] share to be used flexibly by each user.

See also *create mask*, *directory mask*, *force create mode* and *force directory mode*.

Default: **inherit permissions** = no

interfaces (G) — This option allows you to override the default network interfaces list that Samba will use for browsing, name registration and other NBT traffic. By default

Samba will query the kernel for the list of all active interfaces and use any interfaces except 127.0.0.1 that are broadcast capable.

The option takes a list of interface strings. Each string can be in any of the following forms:

- a network interface name (such as eth0). This may include shell-like wildcards so eth* will match any interface starting with the substring "eth"
- an IP address. In this case the netmask is determined from the list of interfaces obtained from the kernel
- an IP/mask pair.
- a broadcast/mask pair.

The "mask" parameters can either be a bit length (such as 24 for a C class network) or a full netmask in dotted decimal form.

The "IP" parameters above can either be a full dotted decimal IP address or a host-name which will be looked up via the OS's normal hostname resolution mechanisms.

For example, the following line:

```
interfaces = eth0 192.168.2.10/24 192.168.3.10/255.255.255.0
```

would configure three network interfaces corresponding to the eth0 device and IP addresses 192.168.2.10 and 192.168.3.10. The netmasks of the latter two interfaces would be set to 255.255.255.0.

See also *bind interfaces only*.

Default: *all active interfaces except 127.0.0.1 that are broadcast capable*

invalid users (S) — This is a list of users that should not be allowed to login to this service. This is really a *paranoid* check to absolutely ensure an improper setting does not breach your security.

A name starting with a '@' is interpreted as an NIS netgroup first (if your system supports NIS), and then as a UNIX group if the name was not found in the NIS netgroup database.

A name starting with '+' is interpreted only by looking in the UNIX group database. A name starting with '&' is interpreted only by looking in the NIS netgroup database (this requires NIS to be working on your system). The characters '+' and '&' may be used at the start of the name in either order so the value *+&group* means check the UNIX group database, followed by the NIS netgroup database, and the value *&+group* means check the NIS netgroup database, followed by the UNIX group database (the same as the '@' prefix).

The current servicename is substituted for *%S*. This is useful in the [homes] section.

See also *valid users*.

Default: *no invalid users*

Example: **invalid users = root fred admin @wheel**

keepalive (G) — The value of the parameter (an integer) represents the number of seconds between *keepalive* packets. If this parameter is zero, no keepalive packets will be sent. Keepalive packets, if sent, allow the server to tell whether a client is still present and responding.

Keepalives should, in general, not be needed if the socket being used has the `SO_KEEPALIVE` attribute set on it (see *socket options*). Basically you should only use this option if you strike difficulties.

Default: **keepalive = 300**

Example: **keepalive = 600**

kernel oplocks (G) — For UNIXes that support kernel based *oplocks* (currently only IRIX and the Linux 2.4 kernel), this parameter allows the use of them to be turned on or off.

Kernel oplocks support allows Samba *oplocks* to be broken whenever a local UNIX process or NFS operation accesses a file that `smbd(8)` has oplocked. This allows complete data consistency between SMB/CIFS, NFS and local file access (and is a *very cool* feature :-).

This parameter defaults to **on**, but is translated to a no-op on systems that do not have the necessary kernel support. You should never need to touch this parameter.

See also the *oplocks* and *level2 oplocks* parameters.

Default: **kernel oplocks = yes**

lanman auth (G) — This parameter determines whether or not `smbd(8)` will attempt to authenticate users using the LANMAN password hash. If disabled, only clients which support NT password hashes (e.g. Windows NT/2000 clients, `smbclient`, etc... but not Windows 95/98 or the MS DOS network client) will be able to connect to the Samba host.

The LANMAN encrypted response is easily broken, due to its case-insensitive nature, and the choice of algorithm. Servers without Windows 95/98 or MS DOS clients are advised to disable this option.

Unlike the **encrypt passwords** option, this parameter cannot alter client behaviour, and the LANMAN response will still be sent over the network. See the **client lanman auth** to disable this for Samba's clients (such as `smbclient`)

If this option, and **ntlm auth** are both disabled, then only NTLMv2 logins will be permitted. Not all clients support NTLMv2, and most will require special configuration to us it.

Default : **lanman auth = yes**

large readwrite (G) — This parameter determines whether or not `smbd(8)` supports the new 64k streaming read and write variant SMB requests introduced with Windows 2000. Note that due to Windows 2000 client redirector bugs this requires Samba to be running on a 64-bit capable operating system such as IRIX, Solaris or a Linux 2.4 kernel. Can improve performance by 10% with Windows 2000 clients. Defaults to `on`. Not as tested as some other Samba code paths.

Default: `large readwrite = yes`

ldap admin dn (G) — The `ldap admin dn` defines the Distinguished Name (DN) name used by Samba to contact the ldap server when retrieving user account information. The `ldap admin dn` is used in conjunction with the admin dn password stored in the `private/secrets.tdb` file. See the `smbpasswd(8)` man page for more information on how to accomplish this.

ldap delete dn (G) — This parameter specifies whether a delete operation in the `ldap-sam` deletes the complete entry or only the attributes specific to Samba.

Default: `ldap delete dn = no`

ldap filter (G) — This parameter specifies the RFC 2254 compliant LDAP search filter. The default is to match the login name with the `uid` attribute for all entries matching the `sambaAccount` objectclass. Note that this filter should only return one entry.

Default: `ldap filter = (&(uid=%u)(objectclass=sambaAccount))`

ldap group suffix (G) — This parameter specifies the suffix that is used for groups when these are added to the LDAP directory. If this parameter is unset, the value of `ldap suffix` will be used instead.

Default: `none`

Example: `dc=samba,ou=Groups`

ldap idmap suffix (G) — This parameter specifies the suffix that is used when storing idmap mappings. If this parameter is unset, the value of `ldap suffix` will be used instead.

Default: `none`

Example: `ou=Idmap,dc=samba,dc=org`

ldap machine suffix (G) — It specifies where machines should be added to the ldap tree.

Default: `none`

ldap passwd sync (G) — This option is used to define whether or not Samba should sync the LDAP password with the NT and LM hashes for normal accounts (NOT for

workstation, server or domain trusts) on a password change via SAMBA.

The *ldap passwd sync* can be set to one of three values:

- *Yes* = Try to update the LDAP, NT and LM passwords and update the `pwd-LastSet` time.
- *No* = Update NT and LM passwords and update the `pwdLastSet` time.
- *Only* = Only update the LDAP password and let the LDAP server do the rest.

Default: **ldap passwd sync = no**

ldap server (G) — This parameter is only available if Samba has been configured to include the `-with-ldapsam` option at compile time.

This parameter should contain the FQDN of the ldap directory server which should be queried to locate user account information.

Default : **ldap server = localhost**

ldap ssl (G) — This option is used to define whether or not Samba should use SSL when connecting to the ldap server. This is *NOT* related to Samba's previous SSL support which was enabled by specifying the `-with-ssl` option to the `configure` script.

The *ldap ssl* can be set to one of three values:

- *Off* = Never use SSL when querying the directory.
- *Start-tls* = Use the LDAPv3 StartTLS extended operation (RFC2830) for communicating with the directory server.
- *On* = Use SSL on the ldaps port when contacting the *ldap server*. Only available when the backwards-compatibility `-with-ldapsam` option is specified to configure. See *passdb backend*

Default : **ldap ssl = start-tls**

ldap suffix (G) — Specifies where user and machine accounts are added to the tree. Can be overridden by **ldap user suffix** and **ldap machine suffix**. It also used as the base dn for all ldap searches.

Default: *none*

ldap user suffix (G) — This parameter specifies where users are added to the tree. If this parameter is not specified, the value from **ldap suffix**.

Default: *none*

level2 oplocks (S) — This parameter controls whether Samba supports level2 (read-only) oplocks on a share.

Level2, or read-only oplocks allow Windows NT clients that have an oplock on a file to downgrade from a read-write oplock to a read-only oplock once a second client opens the file (instead of releasing all oplocks on a second open, as in traditional, exclusive oplocks). This allows all openers of the file that support level2 oplocks to cache the file for read-ahead only (ie. they may not cache writes or lock requests) and increases performance for many accesses of files that are not commonly written (such as application .EXE files).

Once one of the clients which have a read-only oplock writes to the file all clients are notified (no reply is needed or waited for) and told to break their oplocks to "none" and delete any read-ahead caches.

It is recommended that this parameter be turned on to speed access to shared executables.

For more discussions on level2 oplocks see the CIFS spec.

Currently, if *kernel oplocks* are supported then level2 oplocks are not granted (even if this parameter is set to **yes**). Note also, the *oplocks* parameter must be set to **yes** on this share in order for this parameter to have any effect.

See also the *oplocks* and *kernel oplocks* parameters.

Default: **level2 oplocks = yes**

lm announce (G) — This parameter determines if nmbd(8) will produce Lanman announce broadcasts that are needed by OS/2 clients in order for them to see the Samba server in their browse list. This parameter can have three values, **yes**, **no**, or **auto**. The default is **auto**. If set to **no** Samba will never produce these broadcasts. If set to **yes** Samba will produce Lanman announce broadcasts at a frequency set by the parameter *lm interval*. If set to **auto** Samba will not send Lanman announce broadcasts by default but will listen for them. If it hears such a broadcast on the wire it will then start sending them at a frequency set by the parameter *lm interval*.

See also *lm interval*.

Default: **lm announce = auto**

Example: **lm announce = yes**

lm interval (G) — If Samba is set to produce Lanman announce broadcasts needed by OS/2 clients (see the *lm announce* parameter) then this parameter defines the frequency in seconds with which they will be made. If this is set to zero then no Lanman announcements will be made despite the setting of the *lm announce* parameter.

See also *lm announce*.

Default: **lm interval = 60**

Example: **lm interval = 120**

load printers (G) — A boolean variable that controls whether all printers in the printcap will be loaded for browsing by default. See the printers section for more details.

Default: **load printers = yes**

local master (G) — This option allows nmbd(8) to try and become a local master browser on a subnet. If set to **no** then **nmbd** will not attempt to become a local master browser on a subnet and will also lose in all browsing elections. By default this value is set to **yes**. Setting this value to **yes** doesn't mean that Samba will *become* the local master browser on a subnet, just that **nmbd** will *participate* in elections for local master browser.

Setting this value to **no** will cause **nmbd** *never* to become a local master browser.

Default: **local master = yes**

lock dir (G) — Synonym for *lock directory*.

lock directory (G) — This option specifies the directory where lock files will be placed. The lock files are used to implement the *max connections* option.

Default: **lock directory = \${prefix}/var/locks**

Example: **lock directory = /var/run/samba/locks**

locking (S) — This controls whether or not locking will be performed by the server in response to lock requests from the client.

If **locking = no**, all lock and unlock requests will appear to succeed and all lock queries will report that the file in question is available for locking.

If **locking = yes**, real locking will be performed by the server.

This option *may* be useful for read-only filesystems which *may* not need locking (such as CDROM drives), although setting this parameter of **no** is not really recommended even in this case.

Be careful about disabling locking either globally or in a specific service, as lack of locking may result in data corruption. You should never need to set this parameter.

Default: **locking = yes**

lock spin count (G) — This parameter controls the number of times that **smbd** should attempt to gain a byte range lock on the behalf of a client request. Experiments have shown that Windows 2k servers do not reply with a failure if the lock could not be immediately granted, but try a few more times in case the lock could later be aquired. This behavior is used to support PC database formats such as MS Access and FoxPro.

Default: **lock spin count = 3**

lock spin time (G) — The time in microseconds that `smbd` should pause before attempting to gain a failed lock. See *lock spin count* for more details.

Default: **lock spin time = 10**

log file (G) — This option allows you to override the name of the Samba log file (also known as the debug file).

This option takes the standard substitutions, allowing you to have separate log files for each user or machine.

Example: **log file = /usr/local/samba/var/log.%m**

log level (G) — The value of the parameter (a astring) allows the debug level (logging level) to be specified in the `smb.conf` file. This parameter has been extended since the 2.2.x series, now it allow to specify the debug level for multiple debug classes. This is to give greater flexibility in the configuration of the system.

The default will be the log level specified on the command line or level zero if none was specified.

Example: **log level = 3 passdb:5 auth:10 winbind:2**

logon drive (G) — This parameter specifies the local path to which the home directory will be connected (see *logon home*) and is only used by NT Workstations.

Note that this option is only useful if Samba is set up as a logon server.

Default: **logon drive = z:**

Example: **logon drive = h:**

logon home (G) — This parameter specifies the home directory location when a Win95/98 or NT Workstation logs into a Samba PDC. It allows you to do

```
C:\> NET USE H: /HOME
```

from a command prompt, for example.

This option takes the standard substitutions, allowing you to have separate logon scripts for each user or machine.

This parameter can be used with Win9X workstations to ensure that roaming profiles are stored in a subdirectory of the user's home directory. This is done in the following way:

logon home = \\%N\%U\profile

This tells Samba to return the above string, with substitutions made when a client requests the info, generally in a `NetUserGetInfo` request. Win9X clients truncate the info to `\\server\share` when a user does **net use /home** but use the whole string when dealing with profiles.

Note that in prior versions of Samba, the *logon path* was returned rather than *logon home*. This broke `net use /home` but allowed profiles outside the home directory. The current implementation is correct, and can be used for profiles if you use the above trick.

This option is only useful if Samba is set up as a logon server.

Default: `logon home = "\\%N\%U"`

Example: `logon home = "\\remote_smb_server\%U"`

logon path (G) — This parameter specifies the home directory where roaming profiles (NTuser.dat etc files for Windows NT) are stored. Contrary to previous versions of these manual pages, it has nothing to do with Win 9X roaming profiles. To find out how to handle roaming profiles for Win 9X system, see the *logon home* parameter.

This option takes the standard substitutions, allowing you to have separate logon scripts for each user or machine. It also specifies the directory from which the "Application Data", (`desktop`, `start menu`, `network neighborhood`, `programs` and other folders, and their contents, are loaded and displayed on your Windows NT client.

The share and the path must be readable by the user for the preferences and directories to be loaded onto the Windows NT client. The share must be writeable when the user logs in for the first time, in order that the Windows NT client can create the NTuser.dat and other directories.

Thereafter, the directories and any of the contents can, if required, be made read-only. It is not advisable that the NTuser.dat file be made read-only - rename it to NTuser.man to achieve the desired effect (a *MAND*atory profile).

Windows clients can sometimes maintain a connection to the [homes] share, even though there is no user logged in. Therefore, it is vital that the logon path does not include a reference to the homes share (i.e. setting this parameter to `\%N\%U\profile_path` will cause problems).

This option takes the standard substitutions, allowing you to have separate logon scripts for each user or machine.

Note that this option is only useful if Samba is set up as a logon server.

Default: `logon path = "\\%N\%U\profile`

Example: `logon path = "\\PROFILESERVER\PROFILE\%U`

logon script (G) — This parameter specifies the batch file (.bat) or NT command file (.cmd) to be downloaded and run on a machine when a user successfully logs in. The file must contain the DOS style CR/LF line endings. Using a DOS-style editor to create the file is recommended.

The script must be a relative path to the [netlogon] service. If the [netlogon] service specifies a *path* of `/usr/local/samba/netlogon`, and `logon script = STARTUP.BAT`, then the file that will be downloaded is:

```
/usr/local/samba/netlogon/STARTUP.BAT
```

The contents of the batch file are entirely your choice. A suggested command would be to add **NET TIME \\SERVER /SET /YES**, to force every machine to synchronize clocks with the same time server. Another use would be to add **NET USE U: \\SERVER\UTILS** for commonly used utilities, or

```
NET USE Q: \\SERVER\ISO9001_QA
```

for example.

Note that it is particularly important not to allow write access to the [netlogon] share, or to grant users write permission on the batch files in a secure environment, as this would allow the batch files to be arbitrarily modified and security to be breached.

This option takes the standard substitutions, allowing you to have separate logon scripts for each user or machine.

This option is only useful if Samba is set up as a logon server.

Default: *no logon script defined*

Example: **logon script = scripts\%U.bat**

lppause command (S) — This parameter specifies the command to be executed on the server host in order to stop printing or spooling a specific print job.

This command should be a program or script which takes a printer name and job number to pause the print job. One way of implementing this is by using job priorities, where jobs having a too low priority won't be sent to the printer.

If a *%p* is given then the printer name is put in its place. A *%j* is replaced with the job number (an integer). On HPUX (see *printing=hpux*), if the *-p%p* option is added to the lpq command, the job will show up with the correct status, i.e. if the job priority is lower than the set fence priority it will have the PAUSED status, whereas if the priority is equal or higher it will have the SPOOLED or PRINTING status.

Note that it is good practice to include the absolute path in the lppause command as the PATH may not be available to the server.

See also the *printing* parameter.

Default: Currently no default value is given to this string, unless the value of the *printing* parameter is SYSV, in which case the default is :

```
lp -i %p-%j -H hold
```

or if the value of the *printing* parameter is SOFTQ, then the default is:

```
qstat -s -j%j -h
```

Example for HPUX: **lppause command = /usr/bin/lpalt %p-%j -p0**

lpq cache time (G) — This controls how long lpq info will be cached for to prevent the **lpq** command being called too often. A separate cache is kept for each variation of the **lpq** command used by the system, so if you use different **lpq** commands for different users then they won't share cache information.

The cache files are stored in `/tmp/lpq.xxxx` where `xxxx` is a hash of the **lpq** command in use.

The default is 10 seconds, meaning that the cached results of a previous identical **lpq** command will be used if the cached data is less than 10 seconds old. A large value may be advisable if your **lpq** command is very slow.

A value of 0 will disable caching completely.

See also the *printing* parameter.

Default: **lpq cache time = 10**

Example: **lpq cache time = 30**

lpq command (S) — This parameter specifies the command to be executed on the server host in order to obtain **lpq**-style printer status information.

This command should be a program or script which takes a printer name as its only parameter and outputs printer status information.

Currently nine styles of printer status information are supported; BSD, AIX, LPRNG, PLP, SYSV, HPUX, QNX, CUPS, and SOFTQ. This covers most UNIX systems. You control which type is expected using the *printing* = option.

Some clients (notably Windows for Workgroups) may not correctly send the connection number for the printer they are requesting status information about. To get around this, the server reports on the first printer service connected to by the client. This only happens if the connection number sent is invalid.

If a *%p* is given then the printer name is put in its place. Otherwise it is placed at the end of the command.

Note that it is good practice to include the absolute path in the *lpq command* as the `$PATH` may not be available to the server. When compiled with the CUPS libraries, no *lpq command* is needed because `smbd` will make a library call to obtain the print queue listing.

See also the *printing* parameter.

Default: *depends on the setting of printing*

Example: **lpq command = /usr/bin/lpq -P%p**

lpresume command (S) — This parameter specifies the command to be executed on the server host in order to restart or continue printing or spooling a specific print job.

This command should be a program or script which takes a printer name and job number to resume the print job. See also the *lppause command* parameter.

If a *%p* is given then the printer name is put in its place. A *%j* is replaced with the job number (an integer).

Note that it is good practice to include the absolute path in the *lpresume command* as the PATH may not be available to the server.

See also the *printing* parameter.

Default: Currently no default value is given to this string, unless the value of the *printing* parameter is SYSV, in which case the default is :

lp -i %p-%j -H resume

or if the value of the *printing* parameter is SOFTQ, then the default is:

qstat -s -j%j -r

Example for HPUX: **lpresume command = /usr/bin/lpalt %p-%j -p2**

lprm command (S) — This parameter specifies the command to be executed on the server host in order to delete a print job.

This command should be a program or script which takes a printer name and job number, and deletes the print job.

If a *%p* is given then the printer name is put in its place. A *%j* is replaced with the job number (an integer).

Note that it is good practice to include the absolute path in the *lprm command* as the PATH may not be available to the server.

See also the *printing* parameter.

Default: *depends on the setting of printing*

Example 1: **lprm command = /usr/bin/lprm -P%p %j**

Example 2: **lprm command = /usr/bin/cancel %p-%j**

machine password timeout (G) — If a Samba server is a member of a Windows NT Domain (see the *security = domain* parameter) then periodically a running *smbd* process will try and change the MACHINE ACCOUNT PASSWORD stored in the TDB called *private/secrets.tdb*. This parameter specifies how often this password will be changed, in seconds. The default is one week (expressed in seconds), the same as a Windows NT Domain member server.

See also *smbpasswd(8)*, and the *security = domain* parameter.

Default: **machine password timeout = 604800**

magic output (S) — This parameter specifies the name of a file which will contain output created by a magic script (see the *magic script* parameter below).

Warning: If two clients use the same *magic script* in the same directory the output file content is undefined.

Default: **magic output = <magic script name>.out**

Example: **magic output = myfile.txt**

magic script (S) — This parameter specifies the name of a file which, if opened, will be executed by the server when the file is closed. This allows a UNIX script to be sent to the Samba host and executed on behalf of the connected user.

Scripts executed in this way will be deleted upon completion assuming that the user has the appropriate level of privilege and the file permissions allow the deletion.

If the script generates output, output will be sent to the file specified by the *magic output* parameter (see above).

Note that some shells are unable to interpret scripts containing CR/LF instead of CR as the end-of-line marker. Magic scripts must be executable *as is* on the host, which for some hosts and some shells will require filtering at the DOS end.

Magic scripts are *EXPERIMENTAL* and should *NOT* be relied upon.

Default: *None. Magic scripts disabled.*

Example: **magic script = user.csh**

mangle case (S) — See the section on NAME MANGLING

Default: **mangle case = no**

mangled map (S) — This is for those who want to directly map UNIX file names which cannot be represented on Windows/DOS. The mangling of names is not always what is needed. In particular you may have documents with file extensions that differ between DOS and UNIX. For example, under UNIX it is common to use *.html* for HTML files, whereas under Windows/DOS *.htm* is more commonly used.

So to map *html* to *htm* you would use:

mangled map = (*.html *.htm)

One very useful case is to remove the annoying *;1* off the ends of filenames on some CDROMs (only visible under some UNIXes). To do this use a map of *(*;1 *)*.

Default: *no mangled map*

Example: **mangled map = (*;1 *)**

mangled names (S) — This controls whether non-DOS names under UNIX should be mapped to DOS-compatible names ("mangled") and made visible, or whether non-DOS names should simply be ignored.

See the section on NAME MANGLING for details on how to control the mangling process.

If mangling is used then the mangling algorithm is as follows:

- The first (up to) five alphanumeric characters before the rightmost dot of the filename are preserved, forced to upper case, and appear as the first (up to) five characters of the mangled name.
- A tilde "~" is appended to the first part of the mangled name, followed by a two-character unique sequence, based on the original root name (i.e., the original filename minus its final extension). The final extension is included in the hash calculation only if it contains any upper case characters or is longer than three characters.

Note that the character to use may be specified using the *mangling char* option, if you don't like '~'.

- The first three alphanumeric characters of the final extension are preserved, forced to upper case and appear as the extension of the mangled name. The final extension is defined as that part of the original filename after the rightmost dot. If there are no dots in the filename, the mangled name will have no extension (except in the case of "hidden files" - see below).
- Files whose UNIX name begins with a dot will be presented as DOS hidden files. The mangled name will be created as for other filenames, but with the leading dot removed and "___" as its extension regardless of actual original extension (that's three underscores).

The two-digit hash value consists of upper case alphanumeric characters.

This algorithm can cause name collisions only if files in a directory share the same first five alphanumeric characters. The probability of such a clash is 1/1300.

The name mangling (if enabled) allows a file to be copied between UNIX directories from Windows/DOS while retaining the long UNIX filename. UNIX files can be renamed to a new extension from Windows/DOS and will retain the same basename. Mangled names do not change between sessions.

Default: **mangled names = yes**

mangled stack (G) — This parameter controls the number of mangled names that should be cached in the Samba server `smbd(8)`.

This stack is a list of recently mangled base names (extensions are only maintained if they are longer than 3 characters or contains upper case characters).

The larger this value, the more likely it is that mangled names can be successfully converted to correct long UNIX names. However, large stack sizes will slow most directory accesses. Smaller stacks save memory in the server (each stack element costs 256 bytes).

It is not possible to absolutely guarantee correct long filenames, so be prepared for some surprises!

Default: **mangled stack = 50**

Example: **mangled stack = 100**

mangle prefix (G) — controls the number of prefix characters from the original name used when generating the mangled names. A larger value will give a weaker hash and therefore more name collisions. The minimum value is 1 and the maximum value is 6.

mangle prefix is effective only when mangling method is hash2.

Default: **mangle prefix = 1**

Example: **mangle prefix = 4**

mangling char (S) — This controls what character is used as the *magic* character in name mangling. The default is a '~' but this may interfere with some software. Use this option to set it to whatever you prefer. This is effective only when mangling method is hash.

Default: **mangling char = ~**

Example: **mangling char = ^**

mangling method (G) — controls the algorithm used for the generating the mangled names. Can take two different values, "hash" and "hash2". "hash" is the default and is the algorithm that has been used in Samba for many years. "hash2" is a newer and considered a better algorithm (generates less collisions) in the names. However, many Win32 applications store the mangled names and so changing to the new algorithm must not be done lightly as these applications may break unless reinstalled.

Default: **mangling method = hash2**

Example: **mangling method = hash**

map acl inherit (S) — This boolean parameter controls whether `smbd(8)` will attempt to map the 'inherit' and 'protected' access control entry flags stored in Windows ACLs into an extended attribute called `user.SAMBA_PA1`. This parameter only takes effect if Samba is being run on a platform that supports extended attributes (Linux and IRIX so far) and allows the Windows 2000 ACL editor to correctly use inheritance with the Samba POSIX ACL mapping code.

Default: **map acl inherit = no**

map archive (S) — This controls whether the DOS archive attribute should be mapped to the UNIX owner execute bit. The DOS archive bit is set when a file has been modified since its last backup. One motivation for this option is to keep Samba/your PC from making any file it touches from becoming executable under UNIX. This can be quite annoying for shared source code, documents, etc...

Note that this requires the *create mask* parameter to be set such that owner execute bit is not masked out (i.e. it must include 100). See the parameter *create mask* for details.

Default: **map archive = yes**

map hidden (S) — This controls whether DOS style hidden files should be mapped to the UNIX world execute bit.

Note that this requires the *create mask* to be set such that the world execute bit is not masked out (i.e. it must include 001). See the parameter *create mask* for details.

Default: **map hidden = no**

map system (S) — This controls whether DOS style system files should be mapped to the UNIX group execute bit.

Note that this requires the *create mask* to be set such that the group execute bit is not masked out (i.e. it must include 010). See the parameter *create mask* for details.

Default: **map system = no**

map to guest (G) — This parameter is only useful in security modes other than *security = share* - i.e. *user*, *server*, and *domain*.

This parameter can take three different values, which tell `smbd(8)` what to do with user login requests that don't match a valid UNIX user in some way.

The three settings are :

- **Never** - Means user login requests with an invalid password are rejected. This is the default.
- **Bad User** - Means user logins with an invalid password are rejected, unless the username does not exist, in which case it is treated as a guest login and mapped into the *guest account*.
- **Bad Password** - Means user logins with an invalid password are treated as a guest login and mapped into the guest account. Note that this can cause problems as it means that any user incorrectly typing their password will be silently logged on as "guest" - and will not know the reason they cannot access files they think they should - there will have been no message given to them that they got their password wrong. Helpdesk services will *hate* you if you set the *map to guest* parameter this way :-).

Note that this parameter is needed to set up "Guest" share services when using *security* modes other than *share*. This is because in these modes the name of the resource being requested is *not* sent to the server until after the server has successfully authenticated the client so the server cannot make authentication decisions at the correct time (connection to the share) for "Guest" shares.

For people familiar with the older Samba releases, this parameter maps to the old compile-time setting of the `GUEST_SESSSETUP` value in `local.h`.

Default: **map to guest = Never**

Example: **map to guest = Bad User**

max connections (S) — This option allows the number of simultaneous connections to a service to be limited. If *max connections* is greater than 0 then connections will be refused if this number of connections to the service are already open. A value of zero mean an unlimited number of connections may be made.

Record lock files are used to implement this feature. The lock files will be stored in the directory specified by the *lock directory* option.

Default: **max connections = 0**

Example: **max connections = 10**

max disk size (G) — This option allows you to put an upper limit on the apparent size of disks. If you set this option to 100 then all shares will appear to be not larger than 100 MB in size.

Note that this option does not limit the amount of data you can put on the disk. In the above case you could still store much more than 100 MB on the disk, but if a client ever asks for the amount of free disk space or the total disk size then the result will be bounded by the amount specified in *max disk size*.

This option is primarily useful to work around bugs in some pieces of software that can't handle very large disks, particularly disks over 1GB in size.

A *max disk size* of 0 means no limit.

Default: **max disk size = 0**

Example: **max disk size = 1000**

max log size (G) — This option (an integer in kilobytes) specifies the max size the log file should grow to. Samba periodically checks the size and if it is exceeded it will rename the file, adding a *.old* extension.

A size of 0 means no limit.

Default: **max log size = 5000**

Example: **max log size = 1000**

max mux (G) — This option controls the maximum number of outstanding simultaneous SMB operations that Samba tells the client it will allow. You should never need to set this parameter.

Default: **max mux = 50**

max open files (G) — This parameter limits the maximum number of open files that one *smbd(8)* file serving process may have open for a client at any one time. The default for this parameter is set very high (10,000) as Samba uses only one bit per unopened file.

The limit of the number of open files is usually set by the UNIX per-process file descriptor limit rather than this parameter so you should never need to touch this parameter.

Default: **max open files = 10000**

max print jobs (S) — This parameter limits the maximum number of jobs allowable in a Samba printer queue at any given moment. If this number is exceeded, `smbd(8)` will remote "Out of Space" to the client. See all *total print jobs*.

Default: **max print jobs = 1000**

Example: **max print jobs = 5000**

max protocol (G) — The value of the parameter (a string) is the highest protocol level that will be supported by the server.

Possible values are :

- **CORE**: Earliest version. No concept of user names.
- **COREPLUS**: Slight improvements on CORE for efficiency.
- **LANMAN1**: First *modern* version of the protocol. Long filename support.
- **LANMAN2**: Updates to Lanman1 protocol.
- **NT1**: Current up to date version of the protocol. Used by Windows NT. Known as CIFS.

Normally this option should not be set as the automatic negotiation phase in the SMB protocol takes care of choosing the appropriate protocol.

See also *min protocol*

Default: **max protocol = NT1**

Example: **max protocol = LANMAN1**

max reported print jobs (S) — This parameter limits the maximum number of jobs displayed in a port monitor for Samba printer queue at any given moment. If this number is exceeded, the excess jobs will not be shown. A value of zero means there is no limit on the number of print jobs reported. See all *total print jobs* and *max print jobs* parameters.

Default: **max reported print jobs = 0**

Example: **max reported print jobs = 1000**

max smbd processes (G) — This parameter limits the maximum number of `smbd(8)` processes concurrently running on a system and is intended as a stopgap to prevent degrading service to clients in the event that the server has insufficient resources to handle more than this number of connections. Remember that under normal operating

conditions, each user will have an `smbd(8)` associated with him or her to handle connections to all shares from a given host.

Default: **max smbd processes = 0 ##** no limit

Example: **max smbd processes = 1000**

max ttl (G) — This option tells `nmbd(8)` what the default 'time to live' of NetBIOS names should be (in seconds) when `nmbd` is requesting a name using either a broadcast packet or from a WINS server. You should never need to change this parameter. The default is 3 days.

Default: **max ttl = 259200**

max wins ttl (G) — This option tells `smbd(8)` when acting as a WINS server (*wins support = yes*) what the maximum 'time to live' of NetBIOS names that `nmbd` will grant will be (in seconds). You should never need to change this parameter. The default is 6 days (518400 seconds).

See also the *min wins ttl* parameter.

Default: **max wins ttl = 518400**

max xmit (G) — This option controls the maximum packet size that will be negotiated by Samba. The default is 65535, which is the maximum. In some cases you may find you get better performance with a smaller value. A value below 2048 is likely to cause problems.

Default: **max xmit = 65535**

Example: **max xmit = 8192**

message command (G) — This specifies what command to run when the server receives a WinPopup style message.

This would normally be a command that would deliver the message somehow. How this is to be done is up to your imagination.

An example is:

```
message command = csh -c 'xedit %s;rm %s' &
```

This delivers the message using `xedit`, then removes it afterwards. *NOTE THAT IT IS VERY IMPORTANT THAT THIS COMMAND RETURN IMMEDIATELY.* That's why I have the '&' on the end. If it doesn't return immediately then your PCs may freeze when sending messages (they should recover after 30 seconds, hopefully).

All messages are delivered as the global guest user. The command takes the standard substitutions, although `%u` won't work (`%U` may be better in this case).

Apart from the standard substitutions, some additional ones apply. In particular:

- `%s` = the filename containing the message.

- *%t* = the destination that the message was sent to (probably the server name).
- *%f* = who the message is from.

You could make this command send mail, or whatever else takes your fancy. Please let us know of any really interesting ideas you have.

Here's a way of sending the messages as mail to root:

```
message command = /bin/mail -s 'message from %f on %m' root < %s;  
rm %s
```

If you don't have a message command then the message won't be delivered and Samba will tell the sender there was an error. Unfortunately WfWg totally ignores the error code and carries on regardless, saying that the message was delivered.

If you want to silently delete it then try:

```
message command = rm %s
```

Default: *no message command*

Example: **message command = csh -c 'xedit %s; rm %s' &**

min passwd length (G) — Synonym for *min password length*.

min password length (G) — This option sets the minimum length in characters of a plaintext password that **smbd** will accept when performing UNIX password changing.

See also *unix password sync*, *passwd program* and *passwd chat debug*.

Default: **min password length = 5**

min print space (S) — This sets the minimum amount of free disk space that must be available before a user will be able to spool a print job. It is specified in kilobytes. The default is 0, which means a user can always spool a print job.

See also the *printing* parameter.

Default: **min print space = 0**

Example: **min print space = 2000**

min protocol (G) — The value of the parameter (a string) is the lowest SMB protocol dialect than Samba will support. Please refer to the *max protocol* parameter for a list of valid protocol names and a brief description of each. You may also wish to refer to the C source code in `source/smbd/negprot.c` for a listing of known protocol dialects supported by clients.

If you are viewing this parameter as a security measure, you should also refer to the *lanman auth* parameter. Otherwise, you should never need to change this parameter.

Default : **min protocol = CORE**

Example : **min protocol = NT1** # disable DOS clients

min wins ttl (G) — This option tells nmbd(8) when acting as a WINS server (*wins support = yes*) what the minimum 'time to live' of NetBIOS names that **nmbd** will grant will be (in seconds). You should never need to change this parameter. The default is 6 hours (21600 seconds).

Default: **min wins ttl = 21600**

msdfs proxy (S) — This parameter indicates that the share is a stand-in for another CIFS share whose location is specified by the value of the parameter. When clients attempt to connect to this share, they are redirected to the proxied share using the SMB-Dfs protocol.

Only Dfs roots can act as proxy shares. Take a look at the *msdfs root* and *host msdfs* options to find out how to set up a Dfs root share.

Example: **msdfs proxy = \\\\otherserver\\someshare**

msdfs root (S) — If set to **yes**, Samba treats the share as a Dfs root and allows clients to browse the distributed file system tree rooted at the share directory. Dfs links are specified in the share directory by symbolic links of the form **msdfs:serverA\shareA,serverB\shareB** and so on. For more information on setting up a Dfs tree on Samba, refer to Chapter 16, *Hosting a Microsoft Distributed File System Tree*.

See also *host msdfs*

Default: **msdfs root = no**

name cache timeout (G) — Specifies the number of seconds it takes before entries in samba's hostname resolve cache time out. If the timeout is set to 0. the caching is disabled.

Default: **name cache timeout = 660**

Example: **name cache timeout = 0**

name resolve order (G) — This option is used by the programs in the Samba suite to determine what naming services to use and in what order to resolve host names to IP addresses. Its main purpose is to control how netbios name resolution is performed. The option takes a space separated string of name resolution options.

The options are: "lmhosts", "host", "wins" and "bcast". They cause names to be resolved as follows:

- **lmhosts** : Lookup an IP address in the Samba lmhosts file. If the line in lmhosts has no name type attached to the NetBIOS name (see the lmhosts(5) for details) then any name type matches for lookup.

- **host** : Do a standard host name to IP address resolution, using the system `/etc/hosts`, NIS, or DNS lookups. This method of name resolution is operating system depended for instance on IRIX or Solaris this may be controlled by the `/etc/nsswitch.conf` file. Note that this method is used only if the NetBIOS name type being queried is the 0x20 (server) name type or 0x1c (domain controllers). The latter case is only useful for active directory domains and results in a DNS query for the SRV RR entry matching `_ldap._tcp.domain`.
- **wins** : Query a name with the IP address listed in the `wins server` parameter. If no WINS server has been specified this method will be ignored.
- **bcast** : Do a broadcast on each of the known local interfaces listed in the `interfaces` parameter. This is the least reliable of the name resolution methods as it depends on the target host being on a locally connected subnet.

Default: **name resolve order = lmhosts host wins bcast**

Example: **name resolve order = lmhosts bcast host**

This will cause the local `lmhosts` file to be examined first, followed by a broadcast attempt, followed by a normal system `hostname` lookup.

When Samba is functioning in ADS security mode (**security = ads**) it is advised to use following settings for `name resolve order`:

name resolve order = wins bcast

DC lookups will still be done via DNS, but fallbacks to netbios names will not inundate your DNS servers with needless queries for `DOMAIN<0x1c>` lookups.

netbios aliases (G) — This is a list of NetBIOS names that `nmbd` will advertise as additional names by which the Samba server is known. This allows one machine to appear in browse lists under multiple names. If a machine is acting as a browse server or logon server none of these names will be advertised as either browse server or logon servers, only the primary name of the machine will be advertised with these capabilities.

See also `netbios name`.

Default: *empty string (no additional names)*

Example: **netbios aliases = TEST TEST1 TEST2**

netbios name (G) — This sets the NetBIOS name by which a Samba server is known. By default it is the same as the first component of the host's DNS name. If a machine is a browse server or logon server this name (or the first component of the hosts DNS name) will be the name that these services are advertised under.

See also `netbios aliases`.

Default: *machine DNS name*

Example: **netbios name = MYNAME**

netbios scope (G) — This sets the NetBIOS scope that Samba will operate under. This should not be set unless every machine on your LAN also sets this value.

nis homedir (G) — Get the home share server from a NIS map. For UNIX systems that use an automounter, the user's home directory will often be mounted on a workstation on demand from a remote server.

When the Samba logon server is not the actual home directory server, but is mounting the home directories via NFS then two network hops would be required to access the users home directory if the logon server told the client to use itself as the SMB server for home directories (one over SMB and one over NFS). This can be very slow.

This option allows Samba to return the home share as being on a different server to the logon server and as long as a Samba daemon is running on the home directory server, it will be mounted on the Samba client directly from the directory server. When Samba is returning the home share to the client, it will consult the NIS map specified in *homedir map* and return the server listed there.

Note that for this option to work there must be a working NIS system and the Samba server with this option must also be a logon server.

Default: **nis homedir = no**

non unix account range (G) — The non unix account range parameter specifies the range of 'user ids' that are allocated by the various 'non unix account' passdb backends. These backends allow the storage of passwords for users who don't exist in /etc/passwd. This is most often used for machine account creation. This range of ids should have no existing local or NIS users within it as strange conflicts can occur otherwise.

NOTE



These userids never appear on the system and Samba will never 'become' these users. They are used only to ensure that the algorithmic RID mapping does not conflict with normal users.

Default: **non unix account range = <empty string>**

Example: **non unix account range = 10000-20000**

nt acl support (S) — This boolean parameter controls whether `smbd(8)` will attempt to map UNIX permissions into Windows NT access control lists. This parameter was formally a global parameter in releases prior to 2.2.2.

Default: **nt acl support = yes**

ntlm auth (G) — This parameter determines whether or not `smbd(8)` will attempt to authenticate users using the NTLM encrypted password response. If disabled, either the lanman password hash or an NTLMv2 response will need to be sent by the client.

If this option, and **lanman auth** are both disabled, then only NTLMv2 logins will be permitted. Not all clients support NTLMv2, and most will require special configuration to us it.

Default : **ntlm auth = yes**

nt pipe support (G) — This boolean parameter controls whether `smbd(8)` will allow Windows NT clients to connect to the NT SMB specific `IPC$` pipes. This is a developer debugging option and can be left alone.

Default: **nt pipe support = yes**

nt status support (G) — This boolean parameter controls whether `smbd(8)` will negotiate NT specific status support with Windows NT/2k/XP clients. This is a developer debugging option and should be left alone. If this option is set to **no** then Samba offers exactly the same DOS error codes that versions prior to Samba 2.2.3 reported.

You should not need to ever disable this parameter.

Default: **nt status support = yes**

null passwords (G) — Allow or disallow client access to accounts that have null passwords.

See also `smbpasswd(5)`.

Default: **null passwords = no**

obey pam restrictions (G) — When Samba 3.0 is configured to enable PAM support (i.e. `-with-pam`), this parameter will control whether or not Samba should obey PAM's account and session management directives. The default behavior is to use PAM for clear text authentication only and to ignore any account or session management. Note that Samba always ignores PAM for authentication in the case of *encrypt passwords = yes*. The reason is that PAM modules cannot support the challenge/response authentication mechanism needed in the presence of SMB password encryption.

Default: **obey pam restrictions = no**

only guest (S) — A synonym for *guest only*.

only user (S) — This is a boolean option that controls whether connections with usernames not in the *user* list will be allowed. By default this option is disabled so that a client can supply a username to be used by the server. Enabling this parameter will force the server to only use the login names from the *user* list and is only really useful in share level security.

Note that this also means Samba won't try to deduce usernames from the service name. This can be annoying for the [homes] section. To get around this you could use `user = %S` which means your `user` list will be just the service name, which for home directories is the name of the user.

See also the `user` parameter.

Default: `only user = no`

oplock break wait time (G) — This is a tuning parameter added due to bugs in both Windows 9x and WinNT. If Samba responds to a client too quickly when that client issues an SMB that can cause an oplock break request, then the network client can fail and not respond to the break request. This tuning parameter (which is set in milliseconds) is the amount of time Samba will wait before sending an oplock break request to such (broken) clients.

DO NOT CHANGE THIS PARAMETER UNLESS YOU HAVE READ AND UNDERSTOOD THE SAMBA OPLOCK CODE.

Default: `oplock break wait time = 0`

oplock contention limit (S) — This is a *very* advanced `smbd(8)` tuning option to improve the efficiency of the granting of oplocks under multiple client contention for the same file.

In brief it specifies a number, which causes `smbd(8)` not to grant an oplock even when requested if the approximate number of clients contending for an oplock on the same file goes over this limit. This causes `smbd` to behave in a similar way to Windows NT.

DO NOT CHANGE THIS PARAMETER UNLESS YOU HAVE READ AND UNDERSTOOD THE SAMBA OPLOCK CODE.

Default: `oplock contention limit = 2`

oplocks (S) — This boolean option tells `smbd` whether to issue oplocks (opportunistic locks) to file open requests on this share. The oplock code can dramatically (approx. 30% or more) improve the speed of access to files on Samba servers. It allows the clients to aggressively cache files locally and you may want to disable this option for unreliable network environments (it is turned on by default in Windows NT Servers). For more information see the file `Speed.txt` in the Samba `docs/` directory.

Oplocks may be selectively turned off on certain files with a share. See the `veto oplock files` parameter. On some systems oplocks are recognized by the underlying operating system. This allows data synchronization between all access to oplocked files, whether it be via Samba or NFS or a local UNIX process. See the `kernel oplocks` parameter for details.

See also the `kernel oplocks` and `level2 oplocks` parameters.

Default: `oplocks = yes`

os2 driver map (G) — The parameter is used to define the absolute path to a file containing a mapping of Windows NT printer driver names to OS/2 printer driver names. The format is:

```
<nt driver name> = <os2 driver name>.<device name>
```

For example, a valid entry using the HP LaserJet 5 printer driver would appear as **HP LaserJet 5L = LASERJET.HP LaserJet 5L**.

The need for the file is due to the printer driver namespace problem described in Chapter 17, *Classical Printing Support*. For more details on OS/2 clients, please refer to Chapter 37, *Samba and Other CIFS Clients*.

Default: **os2 driver map = <empty string>**

os level (G) — This integer value controls what level Samba advertises itself as for browse elections. The value of this parameter determines whether `nmbd(8)` has a chance of becoming a local master browser for the *WORKGROUP* in the local broadcast area.

Note :By default, Samba will win a local master browsing election over all Microsoft operating systems except a Windows NT 4.0/2000 Domain Controller. This means that a misconfigured Samba host can effectively isolate a subnet for browsing purposes. See `BROWSING.txt` in the Samba `docs/` directory for details.

Default: **os level = 20**

Example: **os level = 65**

pam password change (G) — With the addition of better PAM support in Samba 2.2, this parameter, it is possible to use PAM's password change control flag for Samba. If enabled, then PAM will be used for password changes when requested by an SMB client instead of the program listed in *passwd program*. It should be possible to enable this without changing your *passwd chat* parameter for most setups.

Default: **pam password change = no**

panic action (G) — This is a Samba developer option that allows a system command to be called when either `smbd(8)` or `smbd(8)` crashes. This is usually used to draw attention to the fact that a problem occurred.

Default: **panic action = <empty string>**

Example: **panic action = "/bin/sleep 90000"**

paranoid server security (G) — Some version of NT 4.x allow non-guest users with a bad password. When this option is enabled, samba will not use a broken NT 4.x server as password server, but instead complain to the logs and exit.

Disabling this option prevents Samba from making this check, which involves deliberately attempting a bad logon to the remote server.

Default: **paranoid server security = yes**

passdb backend (G) — This option allows the administrator to choose which backends to retrieve and store passwords with. This allows (for example) both `smbpasswd` and `tdbsam` to be used without a recompile. Multiple backends can be specified, separated by spaces. The backends will be searched in the order they are specified. New users are always added to the first backend specified.

This parameter is in two parts, the backend's name, and a 'location' string that has meaning only to that particular backend. These are separated by a `:` character.

Available backends can include:

- **smbpasswd** - The default `smbpasswd` backend. Takes a path to the `smbpasswd` file as an optional argument.
- **tdbsam** - The TDB based password storage backend. Takes a path to the TDB as an optional argument (defaults to `passdb.tdb` in the *private dir* directory).
- **ldapsam** - The LDAP based `passdb` backend. Takes an LDAP URL as an optional argument (defaults to **ldap://localhost**)

LDAP connections should be secured where possible. This may be done using either Start-TLS (see *ldap ssl*) or by specifying *ldaps://* in the URL argument.

Multiple servers may also be specified in double-quotes, if your LDAP libraries supports the LDAP URL notation. (OpenLDAP does).

- **nisplussam** - The NIS+ based `passdb` backend. Takes name NIS domain as an optional argument. Only works with sun NIS+ servers.
- **mysql** - The MySQL based `passdb` backend. Takes an identifier as argument. Read the Samba HOWTO Collection for configuration details.

Default: **passdb backend = smbpasswd**

Example: **passdb backend = tdbsam:/etc/samba/private/passdb.tdb smbpasswd:/etc/samba/smbpasswd**

Example: **passdb backend = ldapsam:ldaps://ldap.example.com**

Example: **passdb backend = ldapsam:"ldap://ldap-1.example.com ldap://ldap-2.example.com"**

Example: **passdb backend = mysql:my_plugin_args tdbsam**

passwd chat (G) — This string controls the "chat" conversation that takes place between `smbd(8)` and the local password changing program to change the user's password. The string describes a sequence of response-recv pairs that `smbd(8)` uses to determine what to send to the *passwd program* and what to expect back. If the expected output is not received then the password is not changed.

This chat sequence is often quite site specific, depending on what local methods are used for password control (such as NIS etc).

Note that this parameter only is only used if the *unix password sync* parameter is set to **yes**. This sequence is then called *AS ROOT* when the SMB password in the `smbpasswd` file is being changed, without access to the old password cleartext. This means that root must be able to reset the user's password without knowing the text of the previous password. In the presence of NIS/YP, this means that the `passwd` program must be executed on the NIS master.

The string can contain the macro `%n` which is substituted for the new password. The chat sequence can also contain the standard macros `\\n`, `\\r`, `\\t` and `\\s` to give line-feed, carriage-return, tab and space. The chat sequence string can also contain a `*` which matches any sequence of characters. Double quotes can be used to collect strings with spaces in them into a single string.

If the send string in any part of the chat sequence is a full stop `."`, then no string is sent. Similarly, if the expect string is a full stop then no string is expected.

If the *pam password change* parameter is set to **yes**, the chat pairs may be matched in any order, and success is determined by the PAM result, not any particular output. The `\\n` macro is ignored for PAM conversions.

See also *unix password sync*, *passwd program*, *passwd chat debug* and *pam password change*.

Default: `passwd chat = *new*password* %n\\n *new*password* %n\\n *changed*`

Example: `passwd chat = "*Enter OLD password*" %o\\n "*Enter NEW password*" %n\\n "*Reenter NEW password*" %n\\n "*Password changed*"`

passwd chat debug (G) — This boolean specifies if the `passwd chat` script parameter is run in *debug* mode. In this mode the strings passed to and received from the `passwd chat` are printed in the `smbd(8)` log with a *debug level* of 100. This is a dangerous option as it will allow plaintext passwords to be seen in the `smbd` log. It is available to help Samba admins debug their *passwd chat* scripts when calling the *passwd program* and should be turned off after this has been done. This option has no effect if the *pam password change* parameter is set. This parameter is off by default.

See also *passwd chat*, *pam password change*, *passwd program*.

Default: `passwd chat debug = no`

passwd program (G) — The name of a program that can be used to set UNIX user passwords. Any occurrences of `%u` will be replaced with the user name. The user name is checked for existence before calling the password changing program.

Also note that many `passwd` programs insist in *reasonable* passwords, such as a minimum length, or the inclusion of mixed case chars and digits. This can pose a problem

as some clients (such as Windows for Workgroups) uppercase the password before sending it.

Note that if the *unix password sync* parameter is set to **yes** then this program is called *AS ROOT* before the SMB password in the `smbpasswd` file is changed. If this UNIX password change fails, then **smbd** will fail to change the SMB password also (this is by design).

If the *unix password sync* parameter is set this parameter *MUST USE ABSOLUTE PATHS* for *ALL* programs called, and must be examined for security implications. Note that by default *unix password sync* is set to **no**.

See also *unix password sync*.

Default: **passwd program = /bin/passwd**

Example: **passwd program = /sbin/npasswd %u**

password level (G) — Some client/server combinations have difficulty with mixed-case passwords. One offending client is Windows for Workgroups, which for some reason forces passwords to upper case when using the LANMAN1 protocol, but leaves them alone when using COREPLUS! Another problem child is the Windows 95/98 family of operating systems. These clients upper case clear text passwords even when NT LM 0.12 selected by the protocol negotiation request/response.

This parameter defines the maximum number of characters that may be upper case in passwords.

For example, say the password given was "FRED". If *password level* is set to 1, the following combinations would be tried if "FRED" failed:

"Fred", "fred", "fRed", "frEd", "freD"

If *password level* was set to 2, the following combinations would also be tried:

"FRed", "FrEd", "FrED", "fRED", "fReD", "frED", ..

And so on.

The higher value this parameter is set to the more likely it is that a mixed case password will be matched against a single case password. However, you should be aware that use of this parameter reduces security and increases the time taken to process a new connection.

A value of zero will cause only two attempts to be made - the password as is and the password in all-lower case.

Default: **password level = 0**

Example: **password level = 4**

password server (G) — By specifying the name of another SMB server or Active Directory domain controller with this option, and using **security = [ads|domain|server]**

it is possible to get Samba to do all its username/password validation using a specific remote server.

This option sets the name or IP address of the password server to use. New syntax has been added to support defining the port to use when connecting to the server the case of an ADS realm. To define a port other than the default LDAP port of 389, add the port number using a colon after the name or IP address (e.g. 192.168.1.100:389). If you do not specify a port, Samba will use the standard LDAP port of tcp/389. Note that port numbers have no effect on password servers for Windows NT 4.0 domains or netbios connections.

If parameter is a name, it is looked up using the parameter *name resolve order* and so may resolved by any method and order described in that parameter.

The password server must be a machine capable of using the "LM1.2X002" or the "NT LM 0.12" protocol, and it must be in user level security mode.

NOTE



Using a password server means your UNIX box (running Samba) is only as secure as your password server. **DO NOT CHOOSE A PASSWORD SERVER THAT YOU DON'T COMPLETELY TRUST.**

Never point a Samba server at itself for password serving. This will cause a loop and could lock up your Samba server!

The name of the password server takes the standard substitutions, but probably the only useful one is *%m*, which means the Samba server will use the incoming client as the password server. If you use this then you better trust your clients, and you had better restrict them with *hosts allow*!

If the *security* parameter is set to *domain* or *ads*, then the list of machines in this option must be a list of Primary or Backup Domain controllers for the Domain or the character ***, as the Samba server is effectively in that domain, and will use cryptographically authenticated RPC calls to authenticate the user logging on. The advantage of using *security = domain* is that if you list several hosts in the *password server* option then *smbd* will try each in turn till it finds one that responds. This is useful in case your primary server goes down.

If the *password server* option is set to the character ***, then Samba will attempt to auto-locate the Primary or Backup Domain controllers to authenticate against by doing a query for the name *WORKGROUP<1C>* and then contacting each server returned in the list of IP addresses from the name resolution source.

If the list of servers contains both names/IP's and the *** character, the list is treated as a list of preferred domain controllers, but an auto lookup of all remaining DC's will be added to the list as well. Samba will not attempt to optimize this list by locating the closest DC.

If the *security* parameter is set to **server**, then there are different restrictions that **security = domain** doesn't suffer from:

- You may list several password servers in the *password server* parameter, however if an **smbd** makes a connection to a password server, and then the password server fails, no more users will be able to be authenticated from this **smbd**. This is a restriction of the SMB/CIFS protocol when in **security = server** mode and cannot be fixed in Samba.
- If you are using a Windows NT server as your password server then you will have to ensure that your users are able to login from the Samba server, as when in **security = server** mode the network logon will appear to come from there rather than from the users workstation.

See also the *security* parameter.

Default: **password server = <empty string>**

Example: **password server = NT-PDC, NT-BDC1, NT-BDC2, ***

Example: **password server = windc.mydomain.com:389 192.168.1.101 ***

Example: **password server = ***

path (S) — This parameter specifies a directory to which the user of the service is to be given access. In the case of printable services, this is where print data will spool prior to being submitted to the host for printing.

For a printable service offering guest access, the service should be readonly and the path should be world-writeable and have the sticky bit set. This is not mandatory of course, but you probably won't get the results you expect if you do otherwise.

Any occurrences of *%u* in the path will be replaced with the UNIX username that the client is using on this connection. Any occurrences of *%m* will be replaced by the NetBIOS name of the machine they are connecting from. These replacements are very useful for setting up pseudo home directories for users.

Note that this path will be based on *root dir* if one was specified.

Default: *none*

Example: **path = /home/fred**

pid directory (G) — This option specifies the directory where pid files will be placed.

Default: **pid directory = \${prefix}/var/locks**

Example: **pid directory = /var/run/**

posix locking (S) — The **smbd(8)** daemon maintains a database of file locks obtained by SMB clients. The default behavior is to map this internal database to POSIX locks. This means that file locks obtained by SMB clients are consistent with those seen by

POSIX compliant applications accessing the files via a non-SMB method (e.g. NFS or local file access). You should never need to disable this parameter.

Default: **posix locking = yes**

postexec (S) — This option specifies a command to be run whenever the service is disconnected. It takes the usual substitutions. The command may be run as the root on some systems.

An interesting example may be to unmount server resources:

```
postexec = /etc/umount /cdrom
```

See also *preexec*.

Default: *none (no command executed)*

```
Example: postexec = echo \"%u disconnected from %S from %m (%I)\"
>> /tmp/log
```

preexec (S) — This option specifies a command to be run whenever the service is connected to. It takes the usual substitutions.

An interesting example is to send the users a welcome message every time they log in. Maybe a message of the day? Here is an example:

```
preexec = csh -c 'echo \"Welcome to %S!\"
| /usr/local/samba/bin/smbclient -M %m -I %I' &
```

Of course, this could get annoying after a while :-)

See also *preexec close* and *postexec*.

Default: *none (no command executed)*

```
Example: preexec = echo \"%u connected to %S from %m (%I)\" >>
/tmp/log
```

preexec close (S) — This boolean option controls whether a non-zero return code from *preexec* should close the service being connected to.

Default: **preexec close = no**

preferred master (G) — Synonym for *preferred master* for people who cannot spell :-).

preferred master (G) — This boolean parameter controls if nmbd(8) is a preferred master browser for its workgroup.

If this is set to **yes**, on startup, **nmbd** will force an election, and it will have a slight advantage in winning the election. It is recommended that this parameter is used in

conjunction with *domain master = yes*, so that **nmbd** can guarantee becoming a domain master.

Use this option with caution, because if there are several hosts (whether Samba servers, Windows 95 or NT) that are preferred master browsers on the same subnet, they will each periodically and continuously attempt to become the local master browser. This will result in unnecessary broadcast traffic and reduced browsing capabilities.

See also *os level*.

Default: **preferred master = auto**

preload (G) — This is a list of services that you want to be automatically added to the browse lists. This is most useful for homes and printers services that would otherwise not be visible.

Note that if you just want all printers in your printcap file loaded then the *load printers* option is easier.

Default: *no preloaded services*

Example: **preload = fred lp colorlp**

preload modules (G) — This is a list of paths to modules that should be loaded into **smbd** before a client connects. This improves the speed of **smbd** when reacting to new connections somewhat.

Default: **preload modules =**

Example: **preload modules = /usr/lib/samba/passdb/mysql.so+++**

preserve case (S) — This controls if new filenames are created with the case that the client passes, or if they are forced to be the *default case*.

Default: **preserve case = yes**

See the section on NAME MANGLING for a fuller discussion.

printable (S) — If this parameter is **yes**, then clients may open, write to and submit spool files on the directory specified for the service.

Note that a printable service will ALWAYS allow writing to the service path (user privileges permitting) via the spooling of print data. The *read only* parameter controls only non-printing access to the resource.

Default: **printable = no**

printcap (G) — Synonym for *printcap name*.

printcap name (S) — This parameter may be used to override the compiled-in default printcap name used by the server (usually `/etc/printcap`). See the discussion of the [printers] section above for reasons why you might want to do this.

To use the CUPS printing interface set **printcap name = cups**. This should be supplemented by an additional setting `printing = cups` in the [global] section. **printcap name = cups** will use the "dummy" printcap created by CUPS, as specified in your CUPS configuration file.

On System V systems that use **lpstat** to list available printers you can use **printcap name = lpstat** to automatically obtain lists of available printers. This is the default for systems that define SYSV at configure time in Samba (this includes most System V based systems). If *printcap name* is set to **lpstat** on these systems then Samba will launch **lpstat -v** and attempt to parse the output to obtain a printer list.

A minimal printcap file would look something like this:

```
print1|My Printer 1
print2|My Printer 2
print3|My Printer 3
print4|My Printer 4
print5|My Printer 5
```

where the `|` separates aliases of a printer. The fact that the second alias has a space in it gives a hint to Samba that it's a comment.

NOTE



Under AIX the default printcap name is `/etc/qconfig`. Samba will assume the file is in AIX `qconfig` format if the string `qconfig` appears in the printcap filename.

Default: **printcap name = /etc/printcap**

Example: **printcap name = /etc/myprintcap**

print command (S) — After a print job has finished spooling to a service, this command will be used via a **system()** call to process the spool file. Typically the command specified will submit the spool file to the host's printing subsystem, but there is no requirement that this be the case. The server will not remove the spool file, so whatever command you specify should remove the spool file when it has been processed, otherwise you will need to manually remove old spool files.

The `print` command is simply a text string. It will be used verbatim after macro substitutions have been made:

`%s`, `%f` - the path to the spool file name

`%p` - the appropriate printer name

`%J` - the job name as transmitted by the client.

`%c` - The number of printed pages of the spooled job (if known).

`%z` - the size of the spooled print job (in bytes)

The print command *MUST* contain at least one occurrence of `%s` or `%f` - the `%p` is optional. At the time a job is submitted, if no printer name is supplied the `%p` will be silently removed from the printer command.

If specified in the `[global]` section, the print command given will be used for any printable service that does not have its own print command specified.

If there is neither a specified print command for a printable service nor a global print command, spool files will be created but not processed and (most importantly) not removed.

Note that printing may fail on some UNIXes from the `nobody` account. If this happens then create an alternative guest account that can print and set the *guest account* in the `[global]` section.

You can form quite complex print commands by realizing that they are just passed to a shell. For example the following will log a print job, print the file, then remove it. Note that `;` is the usual separator for command in shell scripts.

```
print command = echo Printing %s >> /tmp/print.log; lpr -P %p %s; rm %s
```

You may have to vary this command considerably depending on how you normally print files on your system. The default for the parameter varies depending on the setting of the *printing* parameter.

Default: For **printing = BSD, AIX, QNX, LPRNG or PLP :**

```
print command = lpr -r -P%p %s
```

For **printing = SYSV or HPUX :**

```
print command = lp -c -d%p %s; rm %s
```

For **printing = SOFTQ :**

```
print command = lp -d%p -s %s; rm %s
```

For **printing = CUPS :** If SAMBA is compiled against libcups, then `printcap = cups` uses the CUPS API to submit jobs, etc. Otherwise it maps to the System V commands with the `-oraw` option for printing, i.e. it uses `lp -c -d%p -oraw; rm %s`. With **printing = cups**, and if SAMBA is compiled against libcups, any manually set print command will be ignored.

Example: **print command = /usr/local/samba/bin/myprintscript %p %s**

printer (S) — Synonym for *printer name*.

printer admin (S) — This is a list of users that can do anything to printers via the remote administration interfaces offered by MS-RPC (usually using a NT workstation). Note that the root user always has admin rights.

Default: **printer admin** = <empty string>

Example: **printer admin** = **admin, @staff**

printer name (S) — This parameter specifies the name of the printer to which print jobs spooled through a printable service will be sent.

If specified in the [global] section, the printer name given will be used for any printable service that does not have its own printer name specified.

Default: *none (but may be lp on many systems)*

Example: **printer name** = **laserwriter**

printing (S) — This parameters controls how printer status information is interpreted on your system. It also affects the default values for the *print command*, *lpq command*, *lppause command*, *lpresume command*, and *lprm command* if specified in the [global] section.

Currently nine printing styles are supported. They are BSD, AIX, LPRNG, PLP, SYSV, HPUX, QNX, SOFTQ, and CUPS.

To see what the defaults are for the other print commands when using the various options use the testparm(1) program.

This option can be set on a per printer basis

See also the discussion in the [printers] section.

print ok (S) — Synonym for *printable*.

private dir (G) — This parameters defines the directory smbd will use for storing such files as *smbpasswd* and *secrets.tdb*.

Default :**private dir** = **/\${prefix}/private**

profile acls (S) — This boolean parameter controls whether smbd(8) This boolean parameter was added to fix the problems that people have been having with storing user profiles on Samba shares from Windows 2000 or Windows XP clients. New versions of Windows 2000 or Windows XP service packs do security ACL checking on the owner and ability to write of the profile directory stored on a local workstation when copied from a Samba share.

When not in domain mode with winbindd then the security info copied onto the local workstation has no meaning to the logged in user (SID) on that workstation so

the profile storing fails. Adding this parameter onto a share used for profile storage changes two things about the returned Windows ACL. Firstly it changes the owner and group owner of all reported files and directories to be BUILTIN\\Administrators, BUILTIN\\Users respectively (SIDs S-1-5-32-544, S-1-5-32-545). Secondly it adds an ACE entry of "Full Control" to the SID BUILTIN\\Users to every returned ACL. This will allow any Windows 2000 or XP workstation user to access the profile.

Note that if you have multiple users logging on to a workstation then in order to prevent them from being able to access each others profiles you must remove the "Bypass traverse checking" advanced user right. This will prevent access to other users profile directories as the top level profile directory (named after the user) is created by the workstation profile code and has an ACL restricting entry to the directory tree to the owning user.

Default: **profile acls = no**

protocol (G) — Synonym for *max protocol*.

public (S) — Synonym for *guest ok*.

queuepause command (S) — This parameter specifies the command to be executed on the server host in order to pause the printer queue.

This command should be a program or script which takes a printer name as its only parameter and stops the printer queue, such that no longer jobs are submitted to the printer.

This command is not supported by Windows for Workgroups, but can be issued from the Printers window under Windows 95 and NT.

If a *%p* is given then the printer name is put in its place. Otherwise it is placed at the end of the command.

Note that it is good practice to include the absolute path in the command as the PATH may not be available to the server.

Default: *depends on the setting of printing*

Example: **queuepause command = disable %p**

queueresume command (S) — This parameter specifies the command to be executed on the server host in order to resume the printer queue. It is the command to undo the behavior that is caused by the previous parameter (*queuepause command*).

This command should be a program or script which takes a printer name as its only parameter and resumes the printer queue, such that queued jobs are resubmitted to the printer.

This command is not supported by Windows for Workgroups, but can be issued from the Printers window under Windows 95 and NT.

If a *%p* is given then the printer name is put in its place. Otherwise it is placed at the end of the command.

Note that it is good practice to include the absolute path in the command as the *PATH* may not be available to the server.

Default: *depends on the setting of printing*

Example: **queuepause command = enable %p**

read bmpx (G) — This boolean parameter controls whether *smbd(8)* will support the "Read Block Multiplex" SMB. This is now rarely used and defaults to **no**. You should never need to set this parameter.

Default: **read bmpx = no**

read list (S) — This is a list of users that are given read-only access to a service. If the connecting user is in this list then they will not be given write access, no matter what the *read only* option is set to. The list can include group names using the syntax described in the *invalid users* parameter.

See also the *write list* parameter and the *invalid users* parameter.

Default: **read list = <empty string>**

Example: **read list = mary, @students**

read only (S) — An inverted synonym is *writable*.

If this parameter is **yes**, then users of a service may not create or modify files in the service's directory.

Note that a printable service (**printable = yes**) will *ALWAYS* allow writing to the directory (user privileges permitting), but only via spooling operations.

Default: **read only = yes**

read raw (G) — This parameter controls whether or not the server will support the raw read SMB requests when transferring data to clients.

If enabled, raw reads allow reads of 65535 bytes in one packet. This typically provides a major performance benefit.

However, some clients either negotiate the allowable block size incorrectly or are incapable of supporting larger block sizes, and for these clients you may need to disable raw reads.

In general this parameter should be viewed as a system tuning tool and left severely alone. See also *write raw*.

Default: **read raw = yes**

read size (G) — The option *read size* affects the overlap of disk reads/writes with network reads/writes. If the amount of data being transferred in several of the SMB commands (currently SMBwrite, SMBwriteX and SMBreadbrow) is larger than this value then the server begins writing the data before it has received the whole packet from the network, or in the case of SMBreadbrow, it begins writing to the network before all the data has been read from disk.

This overlapping works best when the speeds of disk and network access are similar, having very little effect when the speed of one is much greater than the other.

The default value is 16384, but very little experimentation has been done yet to determine the optimal value, and it is likely that the best value will vary greatly between systems anyway. A value over 65536 is pointless and will cause you to allocate memory unnecessarily.

Default: **read size = 16384**

Example: **read size = 8192**

realm (G) — This option specifies the kerberos realm to use. The realm is used as the ADS equivalent of the NT4 **domain**. It is usually set to the DNS name of the kerberos server.

Default: **realm =**

Example: **realm = mysambabox.mycompany.com**

remote announce (G) — This option allows you to setup nmbd(8) to periodically announce itself to arbitrary IP addresses with an arbitrary workgroup name.

This is useful if you want your Samba server to appear in a remote workgroup for which the normal browse propagation rules don't work. The remote workgroup can be anywhere that you can send IP packets to.

For example:

remote announce = 192.168.2.255/SERVERS 192.168.4.255/STAFF

the above line would cause **nmbd** to announce itself to the two given IP addresses using the given workgroup names. If you leave out the workgroup name then the one given in the *workgroup* parameter is used instead.

The IP addresses you choose would normally be the broadcast addresses of the remote networks, but can also be the IP addresses of known browse masters if your network config is that stable.

See Chapter 9, *Network Browsing*.

Default: **remote announce = <empty string>**

remote browse sync (G) — This option allows you to setup `nmbd(8)` to periodically request synchronization of browse lists with the master browser of a Samba server that is on a remote segment. This option will allow you to gain browse lists for multiple workgroups across routed networks. This is done in a manner that does not work with any non-Samba servers.

This is useful if you want your Samba server and all local clients to appear in a remote workgroup for which the normal browse propagation rules don't work. The remote workgroup can be anywhere that you can send IP packets to.

For example:

```
remote browse sync = 192.168.2.255 192.168.4.255
```

the above line would cause `nmbd` to request the master browser on the specified subnets or addresses to synchronize their browse lists with the local server.

The IP addresses you choose would normally be the broadcast addresses of the remote networks, but can also be the IP addresses of known browse masters if your network config is that stable. If a machine IP address is given Samba makes NO attempt to validate that the remote machine is available, is listening, nor that it is in fact the browse master on its segment.

Default: **remote browse sync = <empty string>**

restrict anonymous (G) — The setting of this parameter determines whether user and group list information is returned for an anonymous connection. and mirrors the effects of the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA\RestrictAnonymous` registry key in Windows 2000 and Windows NT. When set to 0, user and group list information is returned to anyone who asks. When set to 1, only an authenticated user can retrieve user and group list information. For the value 2, supported by Windows 2000/XP and Samba, no anonymous connections are allowed at all. This can break third party and Microsoft applications which expect to be allowed to perform operations anonymously.

The security advantage of using `restrict anonymous = 1` is dubious, as user and group list information can be obtained using other means.

NOTE



The security advantage of using `restrict anonymous = 2` is removed by setting `guest ok = yes` on any share.

Default: **restrict anonymous = 0**

root (G) — Synonym for `root directory`.

root dir (G) — Synonym for *root directory*".

root directory (G) — The server will **chroot()** (i.e. Change its root directory) to this directory on startup. This is not strictly necessary for secure operation. Even without it the server will deny access to files not in one of the service entries. It may also check for, and deny access to, soft links to other parts of the filesystem, or attempts to use *".."* in file names to access other directories (depending on the setting of the *wide links* parameter).

Adding a *root directory* entry other than *"/* adds an extra level of security, but at a price. It absolutely ensures that no access is given to files not in the sub-tree specified in the *root directory* option, *including* some files needed for complete operation of the server. To maintain full operability of the server you will need to mirror some system files into the *root directory* tree. In particular you will need to mirror */etc/passwd* (or a subset of it), and any binaries or configuration files needed for printing (if required). The set of files that must be mirrored is operating system dependent.

Default: **root directory = /**

Example: **root directory = /homes/smb**

root postexec (S) — This is the same as the *postexec* parameter except that the command is run as root. This is useful for unmounting filesystems (such as CDROMs) after a connection is closed.

See also *postexec*.

Default: **root postexec = <empty string>**

root preexec (S) — This is the same as the *preexec* parameter except that the command is run as root. This is useful for mounting filesystems (such as CDROMs) when a connection is opened.

See also *preexec* and *preexec close*.

Default: **root preexec = <empty string>**

root preexec close (S) — This is the same as the *preexec close* parameter except that the command is run as root.

See also *preexec* and *preexec close*.

Default: **root preexec close = no**

security (G) — This option affects how clients respond to Samba and is one of the most important settings in the *smb.conf* file.

The option sets the "security mode bit" in replies to protocol negotiations with *smbd(8)* to turn share level security on or off. Clients decide based on this bit whether

(and how) to transfer user and password information to the server.

The default is **security = user**, as this is the most common setting needed when talking to Windows 98 and Windows NT.

The alternatives are **security = share**, **security = server** or **security = domain**.

In versions of Samba prior to 2.0.0, the default was **security = share** mainly because that was the only option at one stage.

There is a bug in WfWg that has relevance to this setting. When in user or server level security a WfWg client will totally ignore the password you type in the "connect drive" dialog box. This makes it very difficult (if not impossible) to connect to a Samba service as anyone except the user that you are logged into WfWg as.

If your PCs use usernames that are the same as their usernames on the UNIX machine then you will want to use **security = user**. If you mostly use usernames that don't exist on the UNIX box then use **security = share**.

You should also use **security = share** if you want to mainly setup shares without a password (guest shares). This is commonly used for a shared printer server. It is more difficult to setup guest shares with **security = user**, see the *map to guest* parameter for details.

It is possible to use **smbd** in a *hybrid mode* where it offers both user and share level security under different *NetBIOS aliases*.

The different settings will now be explained.

SECURITY = SHARE

When clients connect to a share level security server they need not log onto the server with a valid username and password before attempting to connect to a shared resource (although modern clients such as Windows 95/98 and Windows NT will send a logon request with a username but no password when talking to a **security = share** server). Instead, the clients send authentication information (passwords) on a per-share basis, at the time they attempt to connect to that share.

Note that **smbd** ALWAYS uses a valid UNIX user to act on behalf of the client, even in **security = share** level security.

As clients are not required to send a username to the server in share level security, **smbd** uses several techniques to determine the correct UNIX user to use on behalf of the client.

A list of possible UNIX usernames to match with the given client password is constructed using the following methods :

- If the *guest only* parameter is set, then all the other stages are missed and only the *guest account* username is checked.
- If a username is sent with the share connection request, then this username (after mapping - see *username map*), is added as a potential username.
- If the client did a previous *logon* request (the SessionSetup SMB call) then the username sent in this SMB will be added as a potential username.

- The name of the service the client requested is added as a potential username.
- The NetBIOS name of the client is added to the list as a potential username.
- Any users on the *user* list are added as potential usernames.

If the *guest only* parameter is not set, then this list is then tried with the supplied password. The first user for whom the password matches will be used as the UNIX user.

If the *guest only* parameter is set, or no username can be determined then if the share is marked as available to the *guest account*, then this guest user will be used, otherwise access is denied.

Note that it can be *very* confusing in share-level security as to which UNIX username will eventually be used in granting access.

See also the section NOTE ABOUT USERNAME/PASSWORD VALIDATION.

SECURITY = USER

This is the default security setting in Samba 3.0. With user-level security a client must first "log-on" with a valid username and password (which can be mapped using the *username map* parameter). Encrypted passwords (see the *encrypted passwords* parameter) can also be used in this security mode. Parameters such as *user* and *guest only* if set are then applied and may change the UNIX user to use on this connection, but only after the user has been successfully authenticated.

Note that the name of the resource being requested is *not* sent to the server until after the server has successfully authenticated the client. This is why guest shares don't work in user level security without allowing the server to automatically map unknown users into the *guest account*. See the *map to guest* parameter for details on doing this.

See also the section NOTE ABOUT USERNAME/PASSWORD VALIDATION.

SECURITY = DOMAIN

This mode will only work correctly if net(8) has been used to add this machine into a Windows NT Domain. It expects the *encrypted passwords* parameter to be set to **yes**. In this mode Samba will try to validate the username/password by passing it to a Windows NT Primary or Backup Domain Controller, in exactly the same way that a Windows NT Server would do.

Note that a valid UNIX user must still exist as well as the account on the Domain Controller to allow Samba to have a valid UNIX account to map file access to.

Note that from the client's point of view **security = domain** is the same as **security = user**. It only affects how the server deals with the authentication, it does not in any way affect what the client sees.

Note that the name of the resource being requested is *not* sent to the server until after the server has successfully authenticated the client. This is why guest shares don't work in user level security without allowing the server to automatically map unknown

users into the *guest account*. See the *map to guest* parameter for details on doing this.

See also the section NOTE ABOUT USERNAME/PASSWORD VALIDATION.

See also the *password server* parameter and the *encrypted passwords* parameter.

SECURITY = SERVER

In this mode Samba will try to validate the username/password by passing it to another SMB server, such as an NT box. If this fails it will revert to **security = user**. It expects the *encrypted passwords* parameter to be set to **yes**, unless the remote server does not support them. However note that if encrypted passwords have been negotiated then Samba cannot revert back to checking the UNIX password file, it must have a valid *smbpasswd* file to check users against. See the chapter about the User Database in the Samba HOWTO Collection for details on how to set this up.

NOTE



This mode of operation has significant pitfalls, due to the fact that it actively initiates a man-in-the-middle attack on the remote SMB server. In particular, this mode of operation can cause significant resource consumption on the PDC, as it must maintain an active connection for the duration of the user's session. Furthermore, if this connection is lost, there is no way to reestablish it, and further authentications to the Samba server may fail. (From a single client, till it disconnects).

NOTE



From the client's point of view **security = server** is the same as **security = user**. It only affects how the server deals with the authentication, it does not in any way affect what the client sees.

Note that the name of the resource being requested is *not* sent to the server until after the server has successfully authenticated the client. This is why guest shares don't work in user level security without allowing the server to automatically map unknown users into the *guest account*. See the *map to guest* parameter for details on doing this.

See also the section NOTE ABOUT USERNAME/PASSWORD VALIDATION.

See also the *password server* parameter and the *encrypted passwords* parameter.

SECURITY = ADS

In this mode, Samba will act as a domain member in an ADS realm. To operate

in this mode, the machine running Samba will need to have Kerberos installed and configured and Samba will need to be joined to the ADS realm using the net utility.

Note that this mode does NOT make Samba operate as a Active Directory Domain Controller.

Read the chapter about Domain Membership in the HOWTO for details.

See also the *realm* paramter and the *encrypted passwords* parameter.

Default: **security = USER**

Example: **security = DOMAIN**

security mask (S) — This parameter controls what UNIX permission bits can be modified when a Windows NT client is manipulating the UNIX permission on a file using the native NT security dialog box.

This parameter is applied as a mask (AND'ed with) to the changed permission bits, thus preventing any bits not in this mask from being modified. Essentially, zero bits in this mask may be treated as a set of bits the user is not allowed to change.

If not set explicitly this parameter is 0777, allowing a user to modify all the user, group or world permissions on a file.

Note that users who can access the Samba server through other means can easily bypass this restriction, so it is primarily useful for standalone "appliance" systems. Administrators of most normal systems will probably want to leave it set to 0777.

See also the *force directory security mode*, *directory security mask*, *force security mode* parameters.

Default: **security mask = 0777**

Example: **security mask = 0770**

server schannel (G) — This controls whether the server offers or even demands the use of the netlogon schannel. *server schannel = no* does not offer the schannel, *server schannel = auto* offers the schannel but does not enforce it, and *server schannel = yes* denies access if the client is not able to speak netlogon schannel. This is only the case for Windows NT4 before SP4.

Please note that with this set to *no* you will have to apply the WindowsXP requireSignOrSeal-Registry patch found in the docs/Registry subdirectory.

Default: **server schannel = auto**

Example: **server schannel = yes**

server signing (G) — This controls whether the server offers or requires the client it talks to to use SMB signing. Possible values are *auto*, *mandatory* and *disabled*.

When set to auto, SMB signing is offered, but not enforced. When set to mandatory, SMB signing is required and if set to disabled, SMB signing is not offered either.

Default: **client signing = Disabled**

server string (G) — This controls what string will show up in the printer comment box in print manager and next to the IPC connection in **net view**. It can be any string that you wish to show to your users.

It also sets what will appear in browse lists next to the machine name.

A *%v* will be replaced with the Samba version number.

A *%h* will be replaced with the hostname.

Default: **server string = Samba %v**

Example: **server string = University of GNUs Samba Server**

set directory (S) — If **set directory = no**, then users of the service may not use the `setdir` command to change directory.

The **setdir** command is only implemented in the Digital Pathworks client. See the Pathworks documentation for details.

Default: **set directory = no**

set primary group script (G) — Thanks to the Posix subsystem in NT a Windows User has a primary group in addition to the auxiliary groups. This script sets the primary group in the unix userdatase when an administrator sets the primary group from the windows user manager or when fetching a SAM with **net rpc vampire**. *%u* will be replaced with the user whose primary group is to be set. *%g* will be replaced with the group to set.

Default: *No default value*

Example: **set primary group script = /usr/sbin/usermod -g '%g' '%u'**

set quota command (G) — The **set quota command** should only be used whenever there is no operating system API available from the OS that samba can use.

This option is only available if Samba was configured with the argument **-with-sys-quotas** or on linux when **./configure -with-quotas** was used and a working quota api was found in the system. Most packages are configured with these options already.

This parameter should specify the path to a script that can set quota for the specified arguments.

The specified script should take the following arguments:

- 1 - quota type
 - 1 - user quotas

- 2 - user default quotas (uid = -1)
- 3 - group quotas
- 4 - group default quotas (gid = -1)
- 2 - id (uid for user, gid for group, -1 if N/A)
- 3 - quota state (0 = disable, 1 = enable, 2 = enable and enforce)
- 4 - block softlimit
- 5 - block hardlimit
- 6 - inode softlimit
- 7 - inode hardlimit
- 8(optional) - block size, defaults to 1024

The script should output at least one line of data on success. And nothing on failure.

See also the *get quota command* parameter.

Default: **set quota command =**

Example: **set quota command = /usr/local/sbin/set_quota**

share modes (S) — This enables or disables the honoring of the *share modes* during a file open. These modes are used by clients to gain exclusive read or write access to a file.

These open modes are not directly supported by UNIX, so they are simulated using shared memory, or lock files if your UNIX doesn't support shared memory (almost all do).

The share modes that are enabled by this option are `DENY_DOS`, `DENY_ALL`, `DENY_READ`, `DENY_WRITE`, `DENY_NONE` and `DENY_FCB`.

This option gives full share compatibility and enabled by default.

You should *NEVER* turn this parameter off as many Windows applications will break if you do so.

Default: **share modes = yes**

short preserve case (S) — This boolean parameter controls if new files which conform to 8.3 syntax, that is all in upper case and of suitable length, are created upper case, or if they are forced to be the *default case*. This option can be use with **preserve case = yes** to permit long filenames to retain their case, while short names are lowered.

See the section on NAME MANGLING.

Default: **short preserve case = yes**

show add printer wizard (G) — With the introduction of MS-RPC based printing support for Windows NT/2000 client in Samba 2.2, a "Printers..." folder will appear on Samba hosts in the share listing. Normally this folder will contain an icon for the MS Add Printer Wizard (APW). However, it is possible to disable this feature regardless of the level of privilege of the connected user.

Under normal circumstances, the Windows NT/2000 client will open a handle on the printer server with `OpenPrinterEx()` asking for Administrator privileges. If the user does not have administrative access on the print server (i.e. is not root or a member of the *printer admin* group), the `OpenPrinterEx()` call fails and the client makes another open call with a request for a lower privilege level. This should succeed, however the APW icon will not be displayed.

Disabling the *show add printer wizard* parameter will always cause the `OpenPrinterEx()` on the server to fail. Thus the APW icon will never be displayed. *Note* :This does not prevent the same user from having administrative privilege on an individual printer.

See also *addprinter command*, *deleteprinter command*, *printer admin*

Default :**show add printer wizard = yes**

shutdown script (G) — *This parameter only exists in the HEAD cvs branch* This a full path name to a script called by `smbd(8)` that should start a shutdown procedure.

This command will be run as the user connected to the server.

`%m %t %r %f` parameters are expanded:

- `%m` will be substituted with the shutdown message sent to the server.
- `%t` will be substituted with the number of seconds to wait before effectively starting the shutdown procedure.
- `%r` will be substituted with the switch *-r*. It means reboot after shutdown for NT.
- `%f` will be substituted with the switch *-f*. It means force the shutdown even if applications do not respond for NT.

Default: *None*.

Example: **shutdown script = /usr/local/samba/sbin/shutdown %m %t %r %f**

Shutdown script example:

```
#!/bin/bash

$time=0
let "time/60"
let "time++"
```

```
/sbin/shutdown $3 $4 +$time $1 &
```

Shutdown does not return so we need to launch it in background.

See also *abort shutdown script*.

smb passwd file (G) — This option sets the path to the encrypted smbpasswd file. By default the path to the smbpasswd file is compiled into Samba.

Default: **smb passwd file = `${prefix}/private/smbpasswd`**

Example: **smb passwd file = `/etc/samba/smbpasswd`**

smb ports (G) — Specifies which ports the server should listen on for SMB traffic.

Default: **smb ports = `445 139`**

socket address (G) — This option allows you to control what address Samba will listen for connections on. This is used to support multiple virtual interfaces on the one server, each with a different configuration.

By default Samba will accept connections on any address.

Example: **socket address = `192.168.2.20`**

socket options (G) — This option allows you to set socket options to be used when talking with the client.

Socket options are controls on the networking layer of the operating systems which allow the connection to be tuned.

This option will typically be used to tune your Samba server for optimal performance for your local network. There is no way that Samba can know what the optimal parameters are for your net, so you must experiment and choose them yourself. We strongly suggest you read the appropriate documentation for your operating system first (perhaps **man setsockopt** will help).

You may find that on some systems Samba will say "Unknown socket option" when you supply an option. This means you either incorrectly typed it or you need to add an include file to includes.h for your OS. If the latter is the case please send the patch to samba-technical@samba.org⁵.

Any of the supported socket options may be combined in any way you like, as long as your OS allows it.

This is the list of socket options currently settable using this option:

- `SO_KEEPALIVE`

⁵<mailto:samba-technical@samba.org>

- `SO_REUSEADDR`
- `SO_BROADCAST`
- `TCP_NODELAY`
- `IPTOS_LOWDELAY`
- `IPTOS_THROUGHPUT`
- `SO_SNDBUF` *
- `SO_RCVBUF` *
- `SO_SNDLOWAT` *
- `SO_RCVLOWAT` *

Those marked with a '*' take an integer argument. The others can optionally take a 1 or 0 argument to enable or disable the option, by default they will be enabled if you don't specify 1 or 0.

To specify an argument use the syntax `SOME_OPTION = VALUE` for example **`SO_SNDBUF = 8192`**. Note that you must not have any spaces before or after the = sign.

If you are on a local network then a sensible option might be:

socket options = IPTOS_LOWDELAY

If you have a local network then you could try:

socket options = IPTOS_LOWDELAY TCP_NODELAY

If you are on a wide area network then perhaps try setting `IPTOS_THROUGHPUT`.

Note that several of the options may cause your Samba server to fail completely. Use these options with caution!

Default: **socket options = TCP_NODELAY**

Example: **socket options = IPTOS_LOWDELAY**

source environment (G) — This parameter causes Samba to set environment variables as per the content of the file named.

If the value of this parameter starts with a "|" character then Samba will treat that value as a pipe command to open and will set the environment variables from the output of the pipe.

The contents of the file or the output of the pipe should be formatted as the output of the standard Unix `env(1)` command. This is of the form:

Example environment entry:

SAMBA_NETBIOS_NAME = myhostname

Default: *No default value*

Examples: **source environment** = `|/etc/smb.conf.sh`

Example: **source environment** = `/usr/local/smb_env_vars`

stat cache (G) — This parameter determines if `smbd(8)` will use a cache in order to speed up case insensitive name mappings. You should never need to change this parameter.

Default: **stat cache** = `yes`

stat cache size (G) — This parameter determines the number of entries in the *stat cache*. You should never need to change this parameter.

Default: **stat cache size** = `50`

strict allocate (S) — This is a boolean that controls the handling of disk space allocation in the server. When this is set to `yes` the server will change from UNIX behaviour of not committing real disk storage blocks when a file is extended to the Windows behaviour of actually forcing the disk system to allocate real storage blocks when a file is created or extended to be a given size. In UNIX terminology this means that Samba will stop creating sparse files. This can be slow on some systems.

When `strict allocate` is `no` the server does sparse disk block allocation when a file is extended.

Setting this to `yes` can help Samba return out of quota messages on systems that are restricting the disk quota of users.

Default: **strict allocate** = `no`

strict locking (S) — This is a boolean that controls the handling of file locking in the server. When this is set to `yes`, the server will check every read and write access for file locks, and deny access if locks exist. This can be slow on some systems.

When `strict locking` is disabled, the server performs file lock checks only when the client explicitly asks for them.

Well-behaved clients always ask for lock checks when it is important. So in the vast majority of cases, **strict locking** = `no` is preferable.

Default: **strict locking** = `no`

strict sync (S) — Many Windows applications (including the Windows 98 explorer shell) seem to confuse flushing buffer contents to disk with doing a `sync` to disk. Under UNIX, a `sync` call forces the process to be suspended until the kernel has ensured that all outstanding data in kernel disk buffers has been safely stored onto stable storage. This is very slow and should only be done rarely. Setting this parameter to `no` (the default) means that `smbd(8)` ignores the Windows applications requests for a `sync` call. There is only a possibility of losing data if the operating system itself that Samba is running on crashes, so there is little danger in this default setting. In addition, this fixes many

performance problems that people have reported with the new Windows98 explorer shell file copies.

See also the *sync always* parameter.

Default: **strict sync = no**

strip dot (G) — This is a boolean that controls whether to strip trailing dots off UNIX filenames. This helps with some CDROMs that have filenames ending in a single dot.

Default: **strip dot = no**

sync always (S) — This is a boolean parameter that controls whether writes will always be written to stable storage before the write call returns. If this is **no** then the server will be guided by the client's request in each write call (clients can set a bit indicating that a particular write should be synchronous). If this is **yes** then every write will be followed by a **fsync()** call to ensure the data is written to disk. Note that the *strict sync* parameter must be set to **yes** in order for this parameter to have any affect.

See also the *strict sync* parameter.

Default: **sync always = no**

syslog (G) — This parameter maps how Samba debug messages are logged onto the system syslog logging levels. Samba debug level zero maps onto syslog LOG_ERR, debug level one maps onto LOG_WARNING, debug level two maps onto LOG_NOTICE, debug level three maps onto LOG_INFO. All higher levels are mapped to LOG_DEBUG.

This parameter sets the threshold for sending messages to syslog. Only messages with debug level less than this value will be sent to syslog.

Default: **syslog = 1**

syslog only (G) — If this parameter is set then Samba debug messages are logged into the system syslog only, and not to the debug log files.

Default: **syslog only = no**

template homedir (G) — When filling out the user information for a Windows NT user, the winbindd(8) daemon uses this parameter to fill in the home directory for that user. If the string *%D* is present it is substituted with the user's Windows NT domain name. If the string *%U* is present it is substituted with the user's Windows NT user name.

Default: **template homedir = /home/%D/%U**

template shell (G) — When filling out the user information for a Windows NT user, the winbindd(8) daemon uses this parameter to fill in the login shell for that user.

Default: **template shell = /bin/false**

time offset (G) — This parameter is a setting in minutes to add to the normal GMT to local time conversion. This is useful if you are serving a lot of PCs that have incorrect daylight saving time handling.

Default: **time offset = 0**

Example: **time offset = 60**

time server (G) — This parameter determines if nmbd(8) advertises itself as a time server to Windows clients.

Default: **time server = no**

timestamp logs (G) — Synonym for *debug timestamp*.

total print jobs (G) — This parameter accepts an integer value which defines a limit on the maximum number of print jobs that will be accepted system wide at any given time. If a print job is submitted by a client which will exceed this number, then smbd(8) will return an error indicating that no space is available on the server. The default value of 0 means that no such limit exists. This parameter can be used to prevent a server from exceeding its capacity and is designed as a printing throttle. See also *max print jobs*.

Default: **total print jobs = 0**

Example: **total print jobs = 5000**

unicode (G) — Specifies whether Samba should try to use unicode on the wire by default. Note: This does NOT mean that samba will assume that the unix machine uses unicode!

Default: **unicode = yes**

unix charset (G) — Specifies the charset the unix machine Samba runs on uses. Samba needs to know this in order to be able to convert text to the charsets other SMB clients use.

Default: **unix charset = UTF8**

Example: **unix charset = ASCII**

unix extensions (G) — This boolean parameter controls whether Samba implements the CIFS UNIX extensions, as defined by HP. These extensions enable Samba to better serve UNIX CIFS clients by supporting features such as symbolic links, hard links, etc... These extensions require a similarly enabled client, and are of no current use to Windows clients.

Default: **unix extensions = yes**

unix password sync (G) — This boolean parameter controls whether Samba attempts to synchronize the UNIX password with the SMB password when the encrypted SMB password in the `smbpasswd` file is changed. If this is set to **yes** the program specified in the `passwd program` parameter is called *AS ROOT* - to allow the new UNIX password to be set without access to the old UNIX password (as the SMB password change code has no access to the old password cleartext, only the new).

See also *passwd program*, *passwd chat*.

Default: **unix password sync = no**

update encrypted (G) — This boolean parameter allows a user logging on with a plaintext password to have their encrypted (hashed) password in the `smbpasswd` file to be updated automatically as they log on. This option allows a site to migrate from plaintext password authentication (users authenticate with plaintext password over the wire, and are checked against a UNIX account database) to encrypted password authentication (the SMB challenge/response authentication mechanism) without forcing all users to re-enter their passwords via `smbpasswd` at the time the change is made. This is a convenience option to allow the change over to encrypted passwords to be made over a longer period. Once all users have encrypted representations of their passwords in the `smbpasswd` file this parameter should be set to **no**.

In order for this parameter to work correctly the *encrypt passwords* parameter must be set to **no** when this parameter is set to **yes**.

Note that even when this parameter is set a user authenticating to **smbd** must still enter a valid password in order to connect correctly, and to update their hashed (`smbpasswd`) passwords.

Default: **update encrypted = no**

use client driver (S) — This parameter applies only to Windows NT/2000 clients. It has no effect on Windows 95/98/ME clients. When serving a printer to Windows NT/2000 clients without first installing a valid printer driver on the Samba host, the client will be required to install a local printer driver. From this point on, the client will treat the print as a local printer and not a network printer connection. This is much the same behavior that will occur when **disable spoolss = yes**.

The differentiating factor is that under normal circumstances, the NT/2000 client will attempt to open the network printer using MS-RPC. The problem is that because the client considers the printer to be local, it will attempt to issue the `OpenPrinterEx()` call requesting access rights associated with the logged on user. If the user possesses local administrator rights but not root privilege on the Samba host (often the case), the `OpenPrinterEx()` call will fail. The result is that the client will now display an "Access Denied; Unable to connect" message in the printer queue window (even though jobs may successfully be printed).

If this parameter is enabled for a printer, then any attempt to open the printer with the `PRINTER_ACCESS_ADMINISTER` right is mapped to `PRINTER_ACCESS_USE`

instead. Thus allowing the `OpenPrinterEx()` call to succeed. *This parameter MUST not be able enabled on a print share which has valid print driver installed on the Samba server.*

See also *disable spoolss*

Default: **use client driver = no**

use mmap (G) — This global parameter determines if the tdb internals of Samba can depend on mmap working correctly on the running system. Samba requires a coherent mmap/read-write system memory cache. Currently only HPUX does not have such a coherent cache, and so this parameter is set to **no** by default on HPUX. On all other systems this parameter should be left alone. This parameter is provided to help the Samba developers track down problems with the tdb internal code.

Default: **use mmap = yes**

user (S) — Synonym for *username*.

username (S) — Multiple users may be specified in a comma-delimited list, in which case the supplied password will be tested against each username in turn (left to right).

The *username* line is needed only when the PC is unable to supply its own username. This is the case for the COREPLUS protocol or where your users have different WfWg usernames to UNIX usernames. In both these cases you may also be better using the `\\server\share%user` syntax instead.

The *username* line is not a great solution in many cases as it means Samba will try to validate the supplied password against each of the usernames in the *username* line in turn. This is slow and a bad idea for lots of users in case of duplicate passwords. You may get timeouts or security breaches using this parameter unwisely.

Samba relies on the underlying UNIX security. This parameter does not restrict who can login, it just offers hints to the Samba server as to what usernames might correspond to the supplied password. Users can login as whoever they please and they will be able to do no more damage than if they started a telnet session. The daemon runs as the user that they log in as, so they cannot do anything that user cannot do.

To restrict a service to a particular set of users you can use the *valid users* parameter.

If any of the usernames begin with a '@' then the name will be looked up first in the NIS netgroups list (if Samba is compiled with netgroup support), followed by a lookup in the UNIX groups database and will expand to a list of all users in the group of that name.

If any of the usernames begin with a '+' then the name will be looked up only in the UNIX groups database and will expand to a list of all users in the group of that name.

If any of the usernames begin with a '&' then the name will be looked up only in the NIS netgroups database (if Samba is compiled with netgroup support) and will

expand to a list of all users in the netgroup group of that name.

Note that searching through a groups database can take quite some time, and some clients may time out during the search.

See the section NOTE ABOUT USERNAME/PASSWORD VALIDATION for more information on how this parameter determines access to the services.

Default: **The guest account if a guest service, else <empty string>.**

Examples: **username = fred, mary, jack, jane, @users, @pcgroup**

username level (G) — This option helps Samba to try and 'guess' at the real UNIX username, as many DOS clients send an all-uppercase username. By default Samba tries all lowercase, followed by the username with the first letter capitalized, and fails if the username is not found on the UNIX machine.

If this parameter is set to non-zero the behavior changes. This parameter is a number that specifies the number of uppercase combinations to try while trying to determine the UNIX user name. The higher the number the more combinations will be tried, but the slower the discovery of usernames will be. Use this parameter when you have strange usernames on your UNIX machine, such as **AstrangeUser**.

Default: **username level = 0**

Example: **username level = 5**

username map (G) — This option allows you to specify a file containing a mapping of usernames from the clients to the server. This can be used for several purposes. The most common is to map usernames that users use on DOS or Windows machines to those that the UNIX box uses. The other is to map multiple users to a single username so that they can more easily share files.

The map file is parsed line by line. Each line should contain a single UNIX username on the left then a '=' followed by a list of usernames on the right. The list of usernames on the right may contain names of the form @group in which case they will match any UNIX username in that group. The special client name '*' is a wildcard and matches any name. Each line of the map file may be up to 1023 characters long.

The file is processed on each line by taking the supplied username and comparing it with each username on the right hand side of the '=' signs. If the supplied name matches any of the names on the right hand side then it is replaced with the name on the left. Processing then continues with the next line.

If any line begins with a '#' or a ';' then it is ignored

If any line begins with an '!' then the processing will stop after that line if a mapping was done by the line. Otherwise mapping continues with every line being processed. Using '!' is most useful when you have a wildcard mapping line later in the file.

For example to map from the name **admin** or **administrator** to the UNIX name **root** you would use:

root = admin administrator

Or to map anyone in the UNIX group `system` to the UNIX name `sys` you would use:

sys = @system

You can have as many mappings as you like in a username map file.

If your system supports the NIS NETGROUP option then the netgroup database is checked before the `/etc/group` database for matching groups.

You can map Windows usernames that have spaces in them by using double quotes around the name. For example:

tridge = "Andrew Tridgell"

would map the windows username "Andrew Tridgell" to the unix username "tridge".

The following example would map mary and fred to the unix user sys, and map the rest to guest. Note the use of the '!' to tell Samba to stop processing if it gets a match on that line.

```
!sys = mary fred
guest = *
```

Note that the remapping is applied to all occurrences of usernames. Thus if you connect to `\\server\fred` and `fred` is remapped to `mary` then you will actually be connecting to `\\server\mary` and will need to supply a password suitable for `mary` not `fred`. The only exception to this is the username passed to the *password server* (if you have one). The password server will receive whatever username the client supplies without modification.

Also note that no reverse mapping is done. The main effect this has is with printing. Users who have been mapped may have trouble deleting print jobs as PrintManager under WfWg will think they don't own the print job.

Default: *no username map*

Example: **username map = /usr/local/samba/lib/users.map**

users (S) — Synonym for *username*.

use sendfile (S) — If this parameter is `yes`, and Samba was built with the `-with-sendfile-support` option, and the underlying operating system supports sendfile system call, then some SMB read calls (mainly `ReadAndX` and `ReadRaw`) will use the more efficient sendfile system call for files that are exclusively oplocked. This may make more efficient use of the system CPU's and cause Samba to be faster. This is off by default as it's effects are unknown as yet.

Default: **use sendfile = no**

use spnego (G) — This variable controls whether samba will try to use Simple and Protected NEGOCIation (as specified by rfc2478) with WindowsXP and Windows2000 clients to agree upon an authentication mechanism. Unless further issues are discovered with our SPNEGO implementation, there is no reason this should ever be disabled.

Default: *use spnego = yes*

utmp (G) — This boolean parameter is only available if Samba has been configured and compiled with the option **–with-utmp**. If set to **yes** then Samba will attempt to add utmp or utmpx records (depending on the UNIX system) whenever a connection is made to a Samba server. Sites may use this to record the user connecting to a Samba share.

Due to the requirements of the utmp record, we are required to create a unique identifier for the incoming user. Enabling this option creates an n^2 algorithm to find this number. This may impede performance on large installations.

See also the *utmp directory* parameter.

Default: **utmp = no**

utmp directory (G) — This parameter is only available if Samba has been configured and compiled with the option **–with-utmp**. It specifies a directory pathname that is used to store the utmp or utmpx files (depending on the UNIX system) that record user connections to a Samba server. See also the *utmp* parameter. By default this is not set, meaning the system will use whatever utmp file the native system is set to use (usually */var/run/utmp* on Linux).

Default: *no utmp directory*

Example: **utmp directory = /var/run/utmp**

-valid (S) — This parameter indicates whether a share is valid and thus can be used. When this parameter is set to false, the share will be in no way visible nor accessible.

This option should not be used by regular users but might be of help to developers. Samba uses this option internally to mark shares as deleted.

Default: *True*

valid users (S) — This is a list of users that should be allowed to login to this service. Names starting with '@', '+' and '&' are interpreted using the same rules as described in the *invalid users* parameter.

If this is empty (the default) then any user can login. If a username is in both this list and the *invalid users* list then access is denied for that user.

The current servicename is substituted for *%S*. This is useful in the [homes] section.

See also *invalid users*

Default: *No valid users list (anyone can login)*

Example: **valid users = greg, @pcusers**

veto files (S) — This is a list of files and directories that are neither visible nor accessible. Each entry in the list must be separated by a '/', which allows spaces to be included in the entry. '*' and '?' can be used to specify multiple files or directories as in DOS wildcards.

Each entry must be a unix path, not a DOS path and must *not* include the unix directory separator '/'.

Note that the *case sensitive* option is applicable in vetoing files.

One feature of the veto files parameter that it is important to be aware of is Samba's behaviour when trying to delete a directory. If a directory that is to be deleted contains nothing but veto files this deletion will *fail* unless you also set the *delete veto files* parameter to *yes*.

Setting this parameter will affect the performance of Samba, as it will be forced to check all files and directories for a match as they are scanned.

See also *hide files* and *case sensitive*.

Default: *No files or directories are vetoed.*

Examples:

```
; Veto any files containing the word Security,
; any ending in .tmp, and any directory containing the
; word root.
veto files = /*Security*/*.tmp/*root*/

; Veto the Apple specific files that a NetAtalk server
; creates.
veto files = /.AppleDouble/.bin/.AppleDesktop/Network Trash Folder/
```

veto oplock files (S) — This parameter is only valid when the *oplocks* parameter is turned on for a share. It allows the Samba administrator to selectively turn off the granting of oplocks on selected files that match a wildcarded list, similar to the wildcarded list used in the *veto files* parameter.

Default: *No files are vetoed for oplock grants*

You might want to do this on files that you know will be heavily contended for by clients. A good example of this is in the NetBench SMB benchmark program, which causes heavy client contention for files ending in *.SEM*. To cause Samba not to grant oplocks on these files you would use the line (either in the [global] section or in the section for the particular NetBench share :

Example: **veto oplock files = /*.SEM/**

vfs object (S) — Synonym for *vfs objects*.

vfs objects (S) — This parameter specifies the backend names which are used for Samba VFS I/O operations. By default, normal disk I/O operations are used but these can be overloaded with one or more VFS objects.

Default: *no value*

Example: **vfs objects = extd_audit recycle**

volume (S) — This allows you to override the volume label returned for a share. Useful for CDROMs with installation programs that insist on a particular volume label.

Default: *the name of the share*

wide links (S) — This parameter controls whether or not links in the UNIX file system may be followed by the server. Links that point to areas within the directory tree exported by the server are always allowed; this parameter controls access only to areas that are outside the directory tree being exported.

Note that setting this parameter can have a negative effect on your server performance due to the extra system calls that Samba has to do in order to perform the link checks.

Default: **wide links = yes**

winbind cache time (G) — This parameter specifies the number of seconds the winbindd(8) daemon will cache user and group information before querying a Windows NT server again.

Default: **winbind cache type = 300**

winbind enum groups (G) — On large installations using winbindd(8) it may be necessary to suppress the enumeration of groups through the **setgrent()**, **getgrent()** and **endgrent()** group of system calls. If the *winbind enum groups* parameter is **no**, calls to the **getgrent()** system call will not return any data.

Warning: Turning off group enumeration may cause some programs to behave oddly.

Default: **winbind enum groups = yes**

winbind enum users (G) — On large installations using winbindd(8) it may be necessary to suppress the enumeration of users through the **setpwent()**, **getpwent()** and **endpwent()** group of system calls. If the *winbind enum users* parameter is **no**, calls to the **getpwent** system call will not return any data.

Warning: Turning off user enumeration may cause some programs to behave oddly. For example, the finger program relies on having access to the full user list when searching for matching usernames.

Default: **winbind enum users = yes**

winbind gid (G) — This parameter is now an alias for **idmap gid**

The winbind gid parameter specifies the range of group ids that are allocated by the winbindd(8) daemon. This range of group ids should have no existing local or NIS groups within it as strange conflicts can occur otherwise.

Default: **winbind gid = <empty string>**

Example: **winbind gid = 10000-20000**

winbind separator (G) — This parameter allows an admin to define the character used when listing a username of the form of *DOMAIN\user*. This parameter is only applicable when using the `pam.winbind.so` and `nss.winbind.so` modules for UNIX services.

Please note that setting this parameter to + causes problems with group membership at least on glibc systems, as the character + is used as a special character for NIS in `/etc/group`.

Default: **winbind separator = '\'**

Example: **winbind separator = +**

winbind uid (G) — This parameter is now an alias for **idmap uid**

The winbind gid parameter specifies the range of user ids that are allocated by the winbindd(8) daemon. This range of ids should have no existing local or NIS users within it as strange conflicts can occur otherwise.

Default: **winbind uid = <empty string>**

Example: **winbind uid = 10000-20000**

winbind use default domain (G) — This parameter specifies whether the winbindd(8) daemon should operate on users without domain component in their username. Users without a domain component are treated as is part of the winbindd server's own domain. While this does not benefit Windows users, it makes SSH, FTP and e-mail function in a way much closer to the way they would in a native unix system.

Default: **winbind use default domain = <no>**

Example: **winbind use default domain = yes**

wins hook (G) — When Samba is running as a WINS server this allows you to call an external program for all changes to the WINS database. The primary use for this option is to allow the dynamic update of external name resolution databases such as dynamic DNS.

The wins hook parameter specifies the name of a script or executable that will be called as follows:

wins.hook operation name nametype ttl IP_list

- The first argument is the operation and is one of "add", "delete", or "refresh". In most cases the operation can be ignored as the rest of the parameters provide sufficient information. Note that "refresh" may sometimes be called when the name has not previously been added, in that case it should be treated as an add.
- The second argument is the NetBIOS name. If the name is not a legal name then the wins hook is not called. Legal names contain only letters, digits, hyphens, underscores and periods.
- The third argument is the NetBIOS name type as a 2 digit hexadecimal number.
- The fourth argument is the TTL (time to live) for the name in seconds.
- The fifth and subsequent arguments are the IP addresses currently registered for that name. If this list is empty then the name should be deleted.

An example script that calls the BIND dynamic DNS update program **nsupdate** is provided in the examples directory of the Samba source code.

wins partners (G) — A space separated list of partners' IP addresses for WINS replication. WINS partners are always defined as push/pull partners as defining only one way WINS replication is unreliable. WINS replication is currently experimental and unreliable between samba servers.

Default: **wins partners =**

Example: **wins partners = 192.168.0.1 172.16.1.2**

wins proxy (G) — This is a boolean that controls if nmbd(8) will respond to broadcast name queries on behalf of other hosts. You may need to set this to **yes** for some older clients.

Default: **wins proxy = no**

wins server (G) — This specifies the IP address (or DNS name: IP address for preference) of the WINS server that nmbd(8) should register with. If you have a WINS server on your network then you should set this to the WINS server's IP.

You should point this at your WINS server if you have a multi-subnetted network.

If you want to work in multiple namespaces, you can give every wins server a 'tag'. For each tag, only one (working) server will be queried for a name. The tag should be separated from the ip address by a colon.

NOTE



You need to set up Samba to point to a WINS server if you have multiple subnets and wish cross-subnet browsing to work correctly.

See the Chapter 9, *Network Browsing*.

Default: *not enabled*

Example: `wins server = mary:192.9.200.1 fred:192.168.3.199 \`
`mary:192.168.2.61`

For this example when querying a certain name, 192.19.200.1 will be asked first and if that doesn't respond 192.168.2.61. If either of those doesn't know the wins name server 192.168.3.199 will be queried.

Example: `wins server = 192.9.200.1 192.168.2.61`

wins support (G) — This boolean controls if the `nmbd(8)` process in Samba will act as a WINS server. You should not set this to `yes` unless you have a multi-subnetted network and you wish a particular `nmbd` to be your WINS server. Note that you should *NEVER* set this to `yes` on more than one machine in your network.

Default: `wins support = no`

workgroup (G) — This controls what workgroup your server will appear to be in when queried by clients. Note that this parameter also controls the Domain name used with the `security = domain` setting.

Default: *set at compile time to WORKGROUP*

Example: `workgroup = MYGROUP`

writable (S) — Synonym for *writeable* for people who can't spell :-).

writeable (S) — Inverted synonym for *read only*.

write cache size (S) — If this integer parameter is set to non-zero value, Samba will create an in-memory cache for each oplocked file (it does *not* do this for non-oplocked files). All writes that the client does not request to be flushed directly to disk will be stored in this cache if possible. The cache is flushed onto disk when a write comes in whose offset would not fit into the cache or when the file is closed by the client. Reads for the file are also served from this cache if the data is stored within it.

This cache allows Samba to batch client writes into a more efficient write size for RAID disks (i.e. writes may be tuned to be the RAID stripe size) and can improve

performance on systems where the disk subsystem is a bottleneck but there is free memory for userspace programs.

The integer parameter specifies the size of this cache (per oplocked file) in bytes.

Default: **write cache size = 0**

Example: **write cache size = 262144**

for a 256k cache size per file.

write list (S) — This is a list of users that are given read-write access to a service. If the connecting user is in this list then they will be given write access, no matter what the *read only* option is set to. The list can include group names using the @group syntax.

Note that if a user is in both the read list and the write list then they will be given write access.

See also the *read list* option.

Default: **write list = <empty string>**

Example: **write list = admin, root, @staff**

write ok (S) — Inverted synonym for *read only*.

write raw (G) — This parameter controls whether or not the server will support raw write SMB's when transferring data from clients. You should never need to change this parameter.

Default: **write raw = yes**

wtmp directory (G) — This parameter is only available if Samba has been configured and compiled with the option **–with-utmp**. It specifies a directory pathname that is used to store the wtmp or wtmpx files (depending on the UNIX system) that record user connections to a Samba server. The difference with the utmp directory is the fact that user info is kept after a user has logged out.

See also the *utmp* parameter. By default this is not set, meaning the system will use whatever utmp file the native system is set to use (usually */var/run/wtmp* on Linux).

Default: *no wtmp directory*

Example: **wtmp directory = /var/log/wtmp**

WARNINGS

Although the configuration file permits service names to contain spaces, your client software may not. Spaces will be ignored in comparisons anyway, so it shouldn't be a problem - but be aware of the possibility.

On a similar note, many clients - especially DOS clients - limit service names to eight characters. `smbd(8)` has no such limitation, but attempts to connect from such clients will fail if they truncate the service names. For this reason you should probably keep your service names down to eight characters in length.

Use of the `[homes]` and `[printers]` special sections make life for an administrator easy, but the various combinations of default attributes can be tricky. Take extreme care when designing these sections. In particular, ensure that the permissions on spool directories are correct.

SEE ALSO

`samba(7)`, `smbpasswd(8)`, `swat(8)`, `smbd(8)`, `nmbd(8)`, `smbclient(1)`, `nmblookup(1)`, `testparm(1)`, `testprns(1)`.

A.2 nmblookup

Synopsis

```
nmblookup [-M] [-R] [-S] [-r] [-A] [-h] [-B <broadcast address>] [-U  
  <unicast address>] [-d <debug level>] [-s <smb config file>] [-i  
  <NetBIOS scope>] [-T] [-f] name
```

DESCRIPTION

This tool is part of the Samba(7) suite.

nmblookup is used to query NetBIOS names and map them to IP addresses in a network using NetBIOS over TCP/IP queries. The options allow the name queries to be directed at a particular IP broadcast area or to a particular machine. All queries are done over UDP.

OPTIONS

- M** — Searches for a master browser by looking up the NetBIOS name *name* with a type of `0x1d`. If *name* is `"-"` then it does a lookup on the special name `_MSBROWSE_`. Please note that in order to use the name `"-"`, you need to make sure `"-"` isn't parsed as an argument, e.g. use : `nmblookup -M -- -`.
- R** — Set the recursion desired bit in the packet to do a recursive lookup. This is used when sending a name query to a machine running a WINS server and the user wishes to query the names in the WINS server. If this bit is unset the normal (broadcast responding) NetBIOS processing code on a machine is used instead. See RFC1001, RFC1002 for details.
- S** — Once the name query has returned an IP address then do a node status query as well. A node status query returns the NetBIOS names registered by a host.

- r** — Try and bind to UDP port 137 to send and receive UDP datagrams. The reason for this option is a bug in Windows 95 where it ignores the source port of the requesting packet and only replies to UDP port 137. Unfortunately, on most UNIX systems root privilege is needed to bind to this port, and in addition, if the `nmbd(8)` daemon is running on this machine it also binds to this port.

- A** — Interpret *name* as an IP Address and do a node status query on this address.

- n** <primary NetBIOS name> — This option allows you to override the NetBIOS name that Samba uses for itself. This is identical to setting the `netbios name` parameter in the `smb.conf` file. However, a command line setting will take precedence over settings in `smb.conf`.

- i** <scope> — This specifies a NetBIOS scope that `nmblookup` will use to communicate with when generating NetBIOS names. For details on the use of NetBIOS scopes, see `rfc1001.txt` and `rfc1002.txt`. NetBIOS scopes are *very* rarely used, only set this parameter if you are the system administrator in charge of all the NetBIOS systems you communicate with.

- W|workgroup=domain** — Set the SMB domain of the username. This overrides the default domain which is the domain defined in `smb.conf`. If the domain specified is the same as the servers NetBIOS name, it causes the client to log on using the servers local SAM (as opposed to the Domain SAM).

- O socket options** — TCP socket options to set on the client socket. See the socket options parameter in the `smb.conf` manual page for the list of valid options.

- h|help** — Print a summary of command line options.

- B** <broadcast address> — Send the query to the given broadcast address. Without this option the default behavior of `nmblookup` is to send the query to the broadcast address of the network interfaces as either auto-detected or defined in the `interfaces`⁶ parameter of the `smb.conf(5)` file.

- U** <unicast address> — Do a unicast query to the specified address or host *unicast address*. This option (along with the `-R` option) is needed to query a WINS server.

- V** — Prints the program version number.

⁶`smb.conf.5.html#INTERFACES`

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|—debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|—logfile=logbasename — File name for log/debug files. The extension `.client` will be appended. The log file is never removed by the client.

-T — This causes any IP addresses found in the lookup to be looked up via a reverse DNS lookup into a DNS name, and printed out before each

IP address NetBIOS name

pair that is the normal output.

-f — Show which flags apply to the name that has been looked up. Possible answers are zero or more of: Response, Authoritative, Truncated, Recursion_Desired, Recursion_Available, Broadcast.

name — This is the NetBIOS name being queried. Depending upon the previous options this may be a NetBIOS name or IP address. If a NetBIOS name then the different name types may be specified by appending `'#<type>'` to the name. This name may also be `'*'`, which will return all registered names within a broadcast area.

EXAMPLES

`nmblookup` can be used to query a WINS server (in the same way `nslookup` is used to query DNS servers). To query a WINS server, `nmblookup` must be called like this:

```
nmblookup -U server -R 'name'
```

For example, running :

```
nmblookup -U samba.org -R 'IRIX#1B'
```

would query the WINS server samba.org for the domain master browser (1B name type) for the IRIX workgroup.

SEE ALSO

nmbd(8), samba(7), and smb.conf(5).

A.3 rpcclient

Synopsis

```
rpcclient [-A authfile] [-c <command string>] [-d debuglevel] [-h] [-l
logfile] [-N] [-s <smb config file>] [-U username[%password]] [-W
workgroup] [-N] [-I destinationIP] server
```

DESCRIPTION

This tool is part of the Samba(7) suite.

rpcclient is a utility initially developed to test MS-RPC functionality in Samba itself. It has undergone several stages of development and stability. Many system administrators have now written scripts around it to manage Windows NT clients from their UNIX workstation.

OPTIONS

server — NetBIOS name of Server to which to connect. The server can be any SMB/CIFS server. The name is resolved using the *name resolve order* line from smb.conf(5).

-c|--command='command string' — execute semicolon separated commands (listed below))

-I IP-address — *IP address* is the address of the server to connect to. It should be specified in standard "a.b.c.d" notation.

Normally the client would attempt to locate a named SMB/CIFS server by looking it up via the NetBIOS name resolution mechanism described above in the *name resolve order* parameter above. Using this parameter will force the client to assume that the server is on the machine with the specified IP address and the NetBIOS name component of the resource being connected to will be ignored.

There is no default for this parameter. If not supplied, it will be determined automatically by the client as described above.

-V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|—debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|—logfile=logbasename — File name for log/debug files. The extension `.client` will be appended. The log file is never removed by the client.

-N — If specified, this parameter suppresses the normal password prompt from the client to the user. This is useful when accessing a service that does not require a password.

Unless a password is specified on the command line or this parameter is specified, the client will request a password.

-k — Try to authenticate with kerberos. Only useful in an Active Directory environment.

-A|—authfile=filename — This option allows you to specify a file from which to read the username and password used in the connection. The format of the file is

```
username = <value>
password = <value>
domain   = <value>
```

Make certain that the permissions on the file restrict access from unwanted users.

-U|—user=username[%password —] Sets the SMB username or username and password.

If `%password` is not specified, the user will be prompted. The client will first check the `USER` environment variable, then the `LOGNAME` variable and if either exists, the string is uppcased. If these environmental variables are not found, the username `GUEST` is used.

A third option is to use a credentials file which contains the plaintext of the username and password. This option is mainly provided for scripts where the admin does not wish to pass the credentials on the command line or via environment variables. If this method is used, make certain that the permissions on the file restrict access from unwanted users. See the `-A` for more details.

Be cautious about including passwords in scripts. Also, on many systems the command line of a running process may be seen via the `ps` command. To be safe always allow `rpcclient` to prompt for a password and type it in directly.

-n <primary NetBIOS name> — This option allows you to override the NetBIOS name that Samba uses for itself. This is identical to setting the `netbios name` parameter in the `smb.conf` file. However, a command line setting will take precedence over settings in `smb.conf`.

-i <scope> — This specifies a NetBIOS scope that `nmblookup` will use to communicate with when generating NetBIOS names. For details on the use of NetBIOS scopes, see `rfc1001.txt` and `rfc1002.txt`. NetBIOS scopes are *very* rarely used, only set this parameter if you are the system administrator in charge of all the NetBIOS systems you communicate with.

-W|`workgroup=domain` — Set the SMB domain of the username. This overrides the default domain which is the domain defined in `smb.conf`. If the domain specified is the same as the servers NetBIOS name, it causes the client to log on using the servers local SAM (as opposed to the Domain SAM).

-O `socket options` — TCP socket options to set on the client socket. See the socket options parameter in the `smb.conf` manual page for the list of valid options.

-h|`help` — Print a summary of command line options.

COMMANDS

LSARPC

`lsaquery` — Query info policy.

`lookupsids` — Resolve a list of SIDs to usernames.

lookupnames — Resolve a list of usernames to SIDs.

enumtrusts — Enumerate trusted domains.

enumprivs — Enumerate privileges.

getdispname — Get the privilege name.

lsaenumsid — Enumerate the LSA SIDS.

lsaenumprivsaccount — Enumerate the privileges of an SID.

lsaenumacctrights — Enumerate the rights of an SID.

lsaenumacctwithright — Enumerate accounts with a right.

lsaaddacctrights — Add rights to an account.

lsaremoveacctrights — Remove rights from an account.

lsalookupprivvalue — Get a privilege value given its name.

lsaquerysecobj — Query LSA security object.

LSARPC-DS

dsroledominfo — Get Primary Domain Information.

DFS

dfsexist — Query DFS support.

dfsadd — Add a DFS share.

dfsremove — Remove a DFS share.

dfsgetinfo — Query DFS share info.

dfsenum — Enumerate dfs shares.

REG

shutdown — Remote Shutdown.

abortshutdown — Abort Shutdown.

SRVSVC

srvinfo — Server query info.

netshareenum — Enumerate shares.

netfileenum — Enumerate open files.

netremotetod — Fetch remote time of day.

SAMR

queryuser — Query user info.

querygroup — Query group info.

queryusergroups — Query user groups.

querygroupmem — Query group membership.

queryaliasmem — Query alias membership.

querydispinfo — Query display info.

querydominfo — Query domain info.

enumdomusers — Enumerate domain users.

enumdomgroups — Enumerate domain groups.

enumalsgroups — Enumerate alias groups.

createdomuser — Create domain user.

samlookupnames — Look up names.

samlookuprids — Look up names.

deletedomuser — Delete domain user.

samquerysecobj — Query SAMR security object.

getdompwinfo — Retrieve domain password info.

lookupdomain — Look up domain.

SPOOLSS

adddriver <arch> <config> — Execute an AddPrinterDriver() RPC to install the printer driver information on the server. Note that the driver files should already exist in the directory returned by **getdriverdir**. Possible values for *arch* are the same as those for the **getdriverdir** command. The *config* parameter is defined as follows:

```

Long Printer Name:\
Driver File Name:\
Data File Name:\
Config File Name:\
Help File Name:\
Language Monitor Name:\
Default Data Type:\
Comma Separated list of Files

```

Any empty fields should be enter as the string "NULL".

Samba does not need to support the concept of Print Monitors since these only apply to local printers whose driver can make use of a bi-directional link for communication. This field should be "NULL". On a remote NT print server, the Print Monitor for a driver must already be installed prior to adding the driver or else the RPC will fail.

addprinter <printername> <sharename> <drivername> <port> — Add a printer on the remote server. This printer will be automatically shared. Be aware

that the printer driver must already be installed on the server (see **adddriver**) and the *port* must be a valid port name (see **enumports**).

deldriver — Delete the specified printer driver for all architectures. This does not delete the actual driver files from the server, only the entry from the server's list of drivers.

enumdata — Enumerate all printer setting data stored on the server. On Windows NT clients, these values are stored in the registry, while Samba servers store them in the printers TDB. This command corresponds to the MS Platform SDK GetPrinterData() function (This command is currently unimplemented.)

enumdataex — Enumerate printer data for a key.

enumjobs <printer> — List the jobs and status of a given printer. This command corresponds to the MS Platform SDK EnumJobs() function.

enumkey — Enumerate printer keys.

enumports [level —] Executes an EnumPorts() call using the specified info level. Currently only info levels 1 and 2 are supported.

enumdrivers [level —] Execute an EnumPrinterDrivers() call. This lists the various installed printer drivers for all architectures. Refer to the MS Platform SDK documentation for more details of the various flags and calling options. Currently supported info levels are 1, 2, and 3.

enumprinters [level —] Execute an EnumPrinters() call. This lists the various installed and share printers. Refer to the MS Platform SDK documentation for more details of the various flags and calling options. Currently supported info levels are 1, 2 and 5.

getdata <printername> <valuename;> — Retrieve the data for a given printer setting. See the **enumdata** command for more information. This command corresponds to the GetPrinterData() MS Platform SDK function.

getdataex — Get printer driver data with keyname.

getdriver <printername> — Retrieve the printer driver information (such as driver file, config file, dependent files, etc...) for the given printer. This command corresponds to the GetPrinterDriver() MS Platform SDK function. Currently info level 1, 2, and 3 are supported.

getdriverdir <arch> — Execute a GetPrinterDriverDirectory() RPC to retrieve the SMB share name and subdirectory for storing printer driver files for a given architecture. Possible values for *arch* are "Windows 4.0" (for Windows 95/98), "Windows NT x86", "Windows NT PowerPC", "Windows Alpha_AXP", and "Windows NT R4000".

getprinter <printername> — Retrieve the current printer information. This command corresponds to the GetPrinter() MS Platform SDK function.

getprintprocdir — Get print processor directory.

openprinter <printername> — Execute an OpenPrinterEx() and ClosePrinter() RPC against a given printer.

setdriver <printername> <drivername> — Execute a SetPrinter() command to update the printer driver associated with an installed printer. The printer driver must already be correctly installed on the print server.

See also the **enumprinters** and **enumdrivers** commands for obtaining a list of installed printers and drivers.

addform — Add form.

setform — Set form.

getform — Get form.

deleteform — Delete form.

enumforms — Enumerate form.

setprinter — Set printer comment.

setprinterdata — Set REG_SZ printer data.

rffpcnex — Rffpcnex test.

NETLOGON

logonctrl2 — Logon Control 2.

logonctrl — Logon Control.

samsync — Sam Synchronisation.

samdeltas — Query Sam Deltas.

samlogon — Sam Logon.

GENERAL COMMANDS

debuglevel — Set the current debug level used to log information.

help (?) — Print a listing of all known commands or extended help on a particular command.

quit (**exit**) — Exit **rpcclient**.

BUGS

rpcclient is designed as a developer testing tool and may not be robust in certain areas (such as command line parsing). It has been known to generate a core dump upon failures when invalid parameters were passed to the interpreter.

From Luke Leighton's original **rpcclient** man page:

WARNING



The MSRPC over SMB code has been developed from examining Network traces. No documentation is available from the original creators (Microsoft) on how MSRPC over SMB works, or how the individual MSRPC services work. Microsoft's implementation of these services has been demonstrated (and reported) to be a bit flaky in places.

The development of Samba's implementation is also a bit rough, and as more of the services are understood, it can even result in versions of **smbd**(8) and **rpcclient**(1) that are incompatible for some commands or services. Additionally, the developers are sending reports to Microsoft, and problems found or reported to Microsoft are fixed in Service Packs, which may result in incompatibilities.

A.4 smbcacls

Synopsis

```
smbcacls //server/share filename [-D acls] [-M acls] [-A acls] [-S acls]
      [-C name] [-G name] [-n] [-t] [-U username] [-h] [-d]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

The **smbcacls** program manipulates NT Access Control Lists (ACLs) on SMB file shares.

OPTIONS

The following options are available to the **smbcacls** program. The format of ACLs is described in the section ACL FORMAT.

- A acls** — Add the ACLs specified to the ACL list. Existing access control entries are unchanged.

- M acls** — Modify the mask value (permissions) for the ACLs specified on the command line. An error will be printed for each ACL specified that was not already present in the ACL list.

- D acls** — Delete any ACLs specified on the command line. An error will be printed for each ACL specified that was not already present in the ACL list.

- S acls** — This command sets the ACLs on the file with only the ones specified on the command line. All other ACLs are erased. Note that the ACL specified must contain at least a revision, type, owner and group for the call to succeed.

- U username** — Specifies a username used to connect to the specified service. The username may be of the form "username" in which case the user is prompted to enter in a password and the workgroup specified in the smb.conf(5) file is used, or "username%password" or "DOMAIN\username%password" and the password and workgroup names are used as provided.

- C name** — The owner of a file or directory can be changed to the name given using the **-C** option. The name can be a sid in the form S-1-x-y-z or a name resolved against the server specified in the first argument.

This command is a shortcut for **-M OWNER:name**.

- G name** — The group owner of a file or directory can be changed to the name given using the **-G** option. The name can be a sid in the form S-1-x-y-z or a name resolved against the server specified in the first argument.

This command is a shortcut for **-M GROUP:name**.

- n** — This option displays all ACL information in numeric format. The default is to convert SIDs to names and ACE types and masks to a readable string format.

-t — Don't actually do anything, only validate the correctness of the arguments.

-h|—help — Print a summary of command line options.

-V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|—debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|—logfile=logbasename — File name for log/debug files. The extension `".client"` will be appended. The log file is never removed by the client.

ACL FORMAT

The format of an ACL is one or more ACL entries separated by either commas or newlines. An ACL entry is one of the following:

```
REVISION:<revision number>
OWNER:<sid or name>
GROUP:<sid or name>
ACL:<sid or name>:<type>/<flags>/<mask>
```

The revision of the ACL specifies the internal Windows NT ACL revision for the security descriptor. If not specified it defaults to 1. Using values other than 1 may cause strange behaviour.

The owner and group specify the owner and group SIDs for the object. If a SID in the format CWS-1-x-y-z is specified this is used, otherwise the name specified is resolved using the server on which the file or directory resides.

ACLs specify permissions granted to the SID. This SID again can be specified in CWS-1-x-y-z format or as a name in which case it is resolved against the server on which the file or directory resides. The type, flags and mask values determine the type of access granted to the SID.

The type can be either 0 or 1 corresponding to ALLOWED or DENIED access to the SID. The flags values are generally zero for file ACLs and either 9 or 2 for directory ACLs. Some common flags are:

- `#define SEC_ACE_FLAG_OBJECT_INHERIT 0x1`
- `#define SEC_ACE_FLAG_CONTAINER_INHERIT 0x2`
- `#define SEC_ACE_FLAG_NO_PROPAGATE_INHERIT 0x4`
- `#define SEC_ACE_FLAG_INHERIT_ONLY 0x8`

At present flags can only be specified as decimal or hexadecimal values.

The mask is a value which expresses the access right granted to the SID. It can be given as a decimal or hexadecimal value, or by using one of the following text strings which map to the NT file permissions of the same name.

- *R* - Allow read access
- *W* - Allow write access
- *X* - Execute permission on the object
- *D* - Delete the object
- *P* - Change permissions
- *O* - Take ownership

The following combined permissions can be specified:

- *READ* - Equivalent to 'RX' permissions
- *CHANGE* - Equivalent to 'RXWD' permissions
- *FULL* - Equivalent to 'RWXDPO' permissions

EXIT STATUS

The `smbcacls` program sets the exit status depending on the success or otherwise of the operations performed. The exit status may be one of the following values.

If the operation succeeded, `smbcacls` returns and exit status of 0. If `smbcacls` couldn't connect to the specified server, or there was an error getting or setting the ACLs, an exit status of 1 is returned. If there was an error parsing any command line arguments, an exit status of 2 is returned.

A.5 smbclient

Synopsis

```
smbclient servicename [password] [-b <buffer size>] [-d debuglevel] [-D
Directory] [-U username] [-W workgroup] [-M <netbios name>] [-m
maxprotocol] [-A authfile] [-N] [-l logfile] [-L <netbios name>] [-I
destinationIP] [-E] [-c <command string>] [-i scope] [-O <socket
options>] [-p port] [-R <name resolve order>] [-s <smb config file>]
[-T<c|x>IXFqgbNan] [-k]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

smbclient is a client that can 'talk' to an SMB/CIFS server. It offers an interface similar to that of the ftp program (see ftp(1)). Operations include things like getting files from the server to the local machine, putting files from the local machine to the server, retrieving directory information from the server and so on.

OPTIONS

servicename — *servicename* is the name of the service you want to use on the server. A service name takes the form *//server/service* where *server* is the NetBIOS name of the SMB/CIFS server offering the desired service and *service* is the name of the service offered. Thus to connect to the service "printer" on the SMB/CIFS server "smbserver", you would use the servicename *//smbserver/printer*

The server name required is NOT necessarily the IP (DNS) host name of the server ! The name required is a NetBIOS server name, which may or may not be the same as the IP hostname of the machine running the server.

The server name is looked up according to either the *-R* parameter to **smbclient** or using the name resolve order parameter in the smb.conf(5) file, allowing an administrator to change the order and methods by which server names are looked up.

password — The password required to access the specified service on the specified server. If this parameter is supplied, the *-N* option (suppress password prompt) is assumed.

There is no default password. If no password is supplied on the command line (either by using this parameter or adding a password to the *-U* option (see below)) and the *-N* option is not specified, the client will prompt for a password, even if the desired service does not require one. (If no password is required, simply press ENTER to provide a null password.)

Some servers (including OS/2 and Windows for Workgroups) insist on an uppercase password. Lowercase or mixed case passwords may be rejected by these servers.

Be cautious about including passwords in scripts.

-R <name resolve order> — This option is used by the programs in the Samba suite to determine what naming services and in what order to resolve host names to IP addresses. The option takes a space-separated string of different name resolution options.

The options are :”lmhosts”, ”host”, ”wins” and ”bcast”. They cause names to be resolved as follows:

- **lmhosts**: Lookup an IP address in the Samba lmhosts file. If the line in lmhosts has no name type attached to the NetBIOS name (see the lmhosts(5) for details) then any name type matches for lookup.
- **host**: Do a standard host name to IP address resolution, using the system /etc/hosts, NIS, or DNS lookups. This method of name resolution is operating system dependent, for instance on IRIX or Solaris this may be controlled by the /etc/nsswitch.conf file). This method is only used if the NetBIOS name type being queried is the 0x20 (server) name type, otherwise it is ignored.
- **wins**: Query a name with the IP address listed in the *wins server* parameter. If no WINS server has been specified this method will be ignored.
- **bcast**: Do a broadcast on each of the known local interfaces listed in the *interfaces* parameter. This is the least reliable of the name resolution methods as it depends on the target host being on a locally connected subnet.

If this parameter is not set then the name resolve order defined in the smb.conf(5) file parameter (name resolve order) will be used.

The default order is lmhosts, host, wins, bcast and without this parameter or any entry in the *name resolve order* parameter of the smb.conf(5) file the name resolution methods will be attempted in this order.

-M NetBIOS name — This options allows you to send messages, using the ”WinPopup” protocol, to another computer. Once a connection is established you then type your message, pressing ^D (control-D) to end.

If the receiving computer is running WinPopup the user will receive the message and probably a beep. If they are not running WinPopup the message will be lost, and no error message will occur.

The message is also automatically truncated if the message is over 1600 bytes, as this is the limit of the protocol.

One useful trick is to cat the message through **smbclient**. For example: **cat mymessage.txt | smbclient -M FRED** will send the message in the file **mymessage.txt** to the machine FRED.

You may also find the **-U** and **-I** options useful, as they allow you to control the FROM and TO parts of the message.

See the *message command* parameter in the smb.conf(5) for a description of how to handle incoming WinPopup messages in Samba.

Copy WinPopup into the startup group on your WfWg PCs if you want them to always be able to receive messages.

-p port — This number is the TCP port number that will be used when making connections to the server. The standard (well-known) TCP port number for an SMB/CIFS server is 139, which is the default.

-l logfilename — If specified, *logfilename* specifies a base filename into which operational data from the running client will be logged.

The default base name is specified at compile time.

The base name is used to generate actual log file names. For example, if the name specified was "log", the debug file would be `log.client`.

The log file generated is never removed by the client.

-h|—help — Print a summary of command line options.

-I IP-address — *IP address* is the address of the server to connect to. It should be specified in standard "a.b.c.d" notation.

Normally the client would attempt to locate a named SMB/CIFS server by looking it up via the NetBIOS name resolution mechanism described above in the *name resolve order* parameter above. Using this parameter will force the client to assume that the server is on the machine with the specified IP address and the NetBIOS name component of the resource being connected to will be ignored.

There is no default for this parameter. If not supplied, it will be determined automatically by the client as described above.

-E — This parameter causes the client to write messages to the standard error stream (stderr) rather than to the standard output stream.

By default, the client writes messages to standard output - typically the user's tty.

-L — This option allows you to look at what services are available on a server. You use it as `smbclient -L host` and a list should appear. The `-I` option may be useful if your NetBIOS names don't match your TCP/IP DNS host names or if you are trying to reach a host on another network.

-t terminal code — This option tells `smbclient` how to interpret filenames coming from the remote server. Usually Asian language multibyte UNIX implementations use different character sets than SMB/CIFS servers (*EUC* instead of *SJIS* for example). Setting this parameter will let `smbclient` convert between the UNIX filenames and the SMB filenames correctly. This option has not been seriously tested and may have some problems.

The terminal codes include CWsjis, CWeuc, CWjis7, CWjis8, CWjunet, CWhex, CW-cap. This is not a complete list, check the Samba source code for the complete list.

-b buffersize — This option changes the transmit/send buffer size when getting or putting a file from/to the server. The default is 65520 bytes. Setting this value smaller (to 1200 bytes) has been observed to speed up file transfers to and from a Win9x server.

-V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|--debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|--logfile=logbasename — File name for log/debug files. The extension `.client` will be appended. The log file is never removed by the client.

-N — If specified, this parameter suppresses the normal password prompt from the client to the user. This is useful when accessing a service that does not require a password.

Unless a password is specified on the command line or this parameter is specified, the client will request a password.

-k — Try to authenticate with kerberos. Only useful in an Active Directory environment.

-A|--authfile=filename — This option allows you to specify a file from which to read the username and password used in the connection. The format of the file is

```
username = <value>
```

```
password = <value>
domain   = <value>
```

Make certain that the permissions on the file restrict access from unwanted users.

-U|**-user=username[%password —]** Sets the SMB username or username and password.

If `%password` is not specified, the user will be prompted. The client will first check the `USER` environment variable, then the `LOGNAME` variable and if either exists, the string is uppercased. If these environmental variables are not found, the username `GUEST` is used.

A third option is to use a credentials file which contains the plaintext of the username and password. This option is mainly provided for scripts where the admin does not wish to pass the credentials on the command line or via environment variables. If this method is used, make certain that the permissions on the file restrict access from unwanted users. See the `-A` for more details.

Be cautious about including passwords in scripts. Also, on many systems the command line of a running process may be seen via the `ps` command. To be safe always allow `rpcclient` to prompt for a password and type it in directly.

-n <primary NetBIOS name> — This option allows you to override the NetBIOS name that Samba uses for itself. This is identical to setting the `netbios name` parameter in the `smb.conf` file. However, a command line setting will take precedence over settings in `smb.conf`.

-i <scope> — This specifies a NetBIOS scope that `nmblookup` will use to communicate with when generating NetBIOS names. For details on the use of NetBIOS scopes, see `rfc1001.txt` and `rfc1002.txt`. NetBIOS scopes are *very* rarely used, only set this parameter if you are the system administrator in charge of all the NetBIOS systems you communicate with.

-W|**-workgroup=domain** — Set the SMB domain of the username. This overrides the default domain which is the domain defined in `smb.conf`. If the domain specified is the same as the servers NetBIOS name, it causes the client to log on using the servers local SAM (as opposed to the Domain SAM).

-O **socket options** — TCP socket options to set on the client socket. See the socket options parameter in the `smb.conf` manual page for the list of valid options.

-T **tar options** — `smbclient` may be used to create **tar(1)** compatible backups of all the files on an SMB/CIFS share. The secondary tar flags that can be given to this option are :

- **c** - Create a tar file on UNIX. Must be followed by the name of a tar file, tape device or `"-"` for standard output. If using standard output you must turn the

log level to its lowest value -d0 to avoid corrupting your tar file. This flag is mutually exclusive with the *x* flag.

- *x* - Extract (restore) a local tar file back to a share. Unless the -D option is given, the tar files will be restored from the top level of the share. Must be followed by the name of the tar file, device or "-" for standard input. Mutually exclusive with the *c* flag. Restored files have their creation times (mtime) set to the date saved in the tar file. Directories currently do not get their creation dates restored properly.
- *I* - Include files and directories. Is the default behavior when filenames are specified above. Causes tar files to be included in an extract or create (and therefore everything else to be excluded). See example below. Filename globbing works in one of two ways. See *r* below.
- *X* - Exclude files and directories. Causes tar files to be excluded from an extract or create. See example below. Filename globbing works in one of two ways now. See *r* below.
- *b* - Blocksize. Must be followed by a valid (greater than zero) blocksize. Causes tar file to be written out in blocksize*TBLOCK (usually 512 byte) blocks.
- *g* - Incremental. Only back up files that have the archive bit set. Useful only with the *c* flag.
- *q* - Quiet. Keeps tar from printing diagnostics as it works. This is the same as tarmode quiet.
- *r* - Regular expression include or exclude. Uses regular expression matching for excluding or excluding files if compiled with HAVE_REGEX_H. However this mode can be very slow. If not compiled with HAVE_REGEX_H, does a limited wildcard match on '*' and '?'.
See *r* below.
- *N* - Newer than. Must be followed by the name of a file whose date is compared against files found on the share during a create. Only files newer than the file specified are backed up to the tar file. Useful only with the *c* flag.
- *a* - Set archive bit. Causes the archive bit to be reset when a file is backed up. Useful with the *g* and *c* flags.

Tar Long File Names

smbclient's tar option now supports long file names both on backup and restore. However, the full path name of the file must be less than 1024 bytes. Also, when a tar archive is created, **smbclient**'s tar option places all files in the archive with relative names, not absolute names.

Tar Filenames

All file names can be given as DOS path names (with '\\\' as the component separator) or as UNIX path names (with '/' as the component separator).

Examples

Restore from tar file **backup.tar** into myshare on mypc (no password on share).

```
smbclient //mypc/yshare "" -N -Tx backup.tar
```

Restore everything except `users/docs`

```
smbclient //mypc/myshare "" -N -TXx backup.tar users/docs
```

Create a tar file of the files beneath `users/docs`.

```
smbclient //mypc/myshare "" -N -Tc backup.tar users/docs
```

Create the same tar file as above, but now use a DOS path name.

```
smbclient //mypc/myshare "" -N -tc backup.tar users\edocs
```

Create a tar file of all the files and directories in the share.

```
smbclient //mypc/myshare "" -N -Tc backup.tar *
```

-D initial directory — Change to initial directory before starting. Probably only of any use with the tar `-T` option.

-c command string — command string is a semicolon-separated list of commands to be executed instead of prompting from stdin. `-N` is implied by `-c`.

This is particularly useful in scripts and for printing stdin to the server, e.g. `-c 'print -'`.

OPERATIONS

Once the client is running, the user is presented with a prompt :

```
smb:\>
```

The backslash ("`\"`") indicates the current working directory on the server, and will change if the current working directory is changed.

The prompt indicates that the client is ready and waiting to carry out a user command. Each command is a single word, optionally followed by parameters specific to that command. Command and parameters are space-delimited unless these notes specifically state otherwise. All commands are case-insensitive. Parameters to commands may or may not be case sensitive, depending on the command.

You can specify file names which have spaces in them by quoting the name with double quotes, for example `"a long file name"`.

Parameters shown in square brackets (e.g., `"[parameter]"`) are optional. If not given, the command will use suitable defaults. Parameters shown in angle brackets (e.g., `"<parameter>"`) are required.

All commands operating on the server are actually performed by issuing a request to the server. Thus the behavior may vary from server to server, depending on how the server was implemented.

The commands available are given here in alphabetical order.

? [**command** —] If *command* is specified, the ? command will display a brief informative message about the specified command. If no command is specified, a list of available commands will be displayed.

! [**shell command** —] If *shell command* is specified, the ! command will execute a shell locally and run the specified shell command. If no command is specified, a local shell will be run.

altname file — The client will request that the server return the "alternate" name (the 8.3 name) for a file or directory.

cancel jobid0 [jobid1 ... [jobidN] —] The client will request that the server cancel the printjobs identified by the given numeric print job ids.

chmod file mode in octal — This command depends on the server supporting the CIFS UNIX extensions and will fail if the server does not. The client requests that the server change the UNIX permissions to the given octal mode, in standard UNIX format.

chown file uid gid — This command depends on the server supporting the CIFS UNIX extensions and will fail if the server does not. The client requests that the server change the UNIX user and group ownership to the given decimal values. There is currently no way to remotely look up the UNIX UID and GID values for a given name. This may be addressed in future versions of the CIFS UNIX extensions.

cd [directory name —] If "directory name" is specified, the current working directory on the server will be changed to the directory specified. This operation will fail if for any reason the specified directory is inaccessible.

If no directory name is specified, the current working directory on the server will be reported.

del <mask> — The client will request that the server attempt to delete all files matching *mask* from the current working directory on the server.

dir <mask> — A list of the files matching *mask* in the current working directory on the server will be retrieved from the server and displayed.

exit — Terminate the connection with the server and exit from the program.

get <remote file name> [local file name —] Copy the file called *remote file name* from the server to the machine running the client. If specified, name the local copy *local file name*. All transfers in **smbclient** are binary. See also the lowercase command.

help [command —] See the ? command above.

lcd [directory name —] If *directory name* is specified, the current working directory on the local machine will be changed to the directory specified. This operation will

fail if for any reason the specified directory is inaccessible.

If no directory name is specified, the name of the current working directory on the local machine will be reported.

link source destination — This command depends on the server supporting the CIFS UNIX extensions and will fail if the server does not. The client requests that the server create a hard link between the source and destination files. The source file must not exist.

lowercase — Toggle lowercasing of filenames for the get and mget commands.

When lowercasing is toggled ON, local filenames are converted to lowercase when using the get and mget commands. This is often useful when copying (say) MSDOS files from a server, because lowercase filenames are the norm on UNIX systems.

ls <mask> — See the dir command above.

mask <mask> — This command allows the user to set up a mask which will be used during recursive operation of the mget and mput commands.

The masks specified to the mget and mput commands act as filters for directories rather than files when recursion is toggled ON.

The mask specified with the mask command is necessary to filter files within those directories. For example, if the mask specified in an mget command is "source*" and the mask specified with the mask command is "*.c" and recursion is toggled ON, the mget command will retrieve all files matching "*.c" in all directories below and including all directories matching "source*" in the current working directory.

The value for mask defaults to blank (equivalent to "") and remains so until the mask command is used to change it. It retains the most recently specified value indefinitely. To avoid unexpected results it would be wise to change the value of mask back to "" after using the mget or mput commands.

md <directory name> — See the mkdir command.

mget <mask> — Copy all files matching *mask* from the server to the machine running the client.

mask is interpreted differently during recursive operation and non-recursive operation - refer to the recurse and mask commands for more information. All transfers in **smbclient** are binary. See also the lowercase command.

mkdir <directory name> — Create a new directory on the server (user access privileges permitting) with the specified name.

mput <**mask**> — Copy all files matching *mask* in the current working directory on the local machine to the current working directory on the server.

mask is interpreted differently during recursive operation and non-recursive operation - refer to the `recurse` and `mask` commands for more information. All transfers in **smbclient** are binary.

print <**file name**> — Print the specified file from the local machine through a printable service on the server.

See also the `printmode` command.

printmode <**graphics or text**> — Set the print mode to suit either binary data (such as graphical information) or text. Subsequent print commands will use the currently set print mode.

prompt — Toggle prompting for filenames during operation of the `mget` and `mput` commands.

When toggled ON, the user will be prompted to confirm the transfer of each file during these commands. When toggled OFF, all specified files will be transferred without prompting.

put <**local file name**> [**remote file name** —] Copy the file called **local file name** from the machine running the client to the server. If specified, name the remote copy **remote file name**. All transfers in **smbclient** are binary. See also the lowercase command.

queue — Displays the print queue, showing the job id, name, size and current status.

quit — See the `exit` command.

rd <**directory name**> — See the `rmdir` command.

recurse — Toggle directory recursion for the commands `mget` and `mput`.

When toggled ON, these commands will process all directories in the source directory (i.e., the directory they are copying from) and will recurse into any that match the mask specified to the command. Only files that match the mask specified using the `mask` command will be retrieved. See also the `mask` command.

When recursion is toggled OFF, only files from the current working directory on the source machine that match the mask specified to the `mget` or `mput` commands will be copied, and any mask specified using the `mask` command will be ignored.

rm <**mask**> — Remove all files matching *mask* from the current working directory on the server.

rmdir <directory name> — Remove the specified directory (user access privileges permitting) from the server.

setmode <filename> <perm=[+|\- rsha> —] A version of the DOS attrib command to set file permissions. For example:

```
setmode myfile +r
```

would make myfile read only.

symlink source destination — This command depends on the server supporting the CIFS UNIX extensions and will fail if the server does not. The client requests that the server create a symbolic hard link between the source and destination files. The source file must not exist. The server will not create a link to any path that lies outside the currently connected share. This is enforced by the Samba server.

tar <c|x>[IXbgNa —] Performs a tar operation - see the *-T* command line option above. Behavior may be affected by the *tarmode* command (see below). Using *g* (incremental) and *N* (newer) will affect *tarmode* settings. Using the *"-"* option with *tar x* may not work - use the command line option instead.

blocksize <blocksize> — Blocksize. Must be followed by a valid (greater than zero) blocksize. Causes tar file to be written out in *blocksize**TBLOCK (usually 512 byte) blocks.

tarmode <full|inc|reset|noreset> — Changes tar's behavior with regard to archive bits. In full mode, tar will back up everything regardless of the archive bit setting (this is the default mode). In incremental mode, tar will only back up files with the archive bit set. In reset mode, tar will reset the archive bit on all files it backs up (implies read/write share).

NOTES

Some servers are fussy about the case of supplied usernames, passwords, share names (AKA service names) and machine names. If you fail to connect try giving all parameters in uppercase.

It is often necessary to use the *-n* option when connecting to some types of servers. For example OS/2 LanManager insists on a valid NetBIOS name being used, so you need to supply a valid name that would be known to the server.

smbclient supports long file names where the server supports the LANMAN2 protocol or above.

ENVIRONMENT VARIABLES

The variable `USER` may contain the username of the person using the client. This information is used only if the protocol level is high enough to support session-level passwords.

The variable `PASSWD` may contain the password of the person using the client. This information is used only if the protocol level is high enough to support session-level passwords.

The variable `LIBSMB_PROG` may contain the path, executed with `system()`, which the client should connect to instead of connecting to a server. This functionality is primarily intended as a development aid, and works best when using a `LMHOSTS` file

INSTALLATION

The location of the client program is a matter for individual system administrators. The following are thus suggestions only.

It is recommended that the `smbclient` software be installed in the `/usr/local/samba/bin/` or `/usr/samba/bin/` directory, this directory readable by all, writeable only by root. The client program itself should be executable by all. The client should *NOT* be `setuid` or `setgid`!

The client log files should be put in a directory readable and writeable only by the user.

To test the client, you will need to know the name of a running SMB/CIFS server. It is possible to run `smbd(8)` as an ordinary user - running that server as a daemon on a user-accessible port (typically any port number over 1024) would provide a suitable test server.

DIAGNOSTICS

Most diagnostics issued by the client are logged in a specified log file. The log file name is specified at compile time, but may be overridden on the command line.

The number and nature of diagnostics available depends on the debug level used by the client. If you have problems, set the debug level to 3 and peruse the log files.

A.6 net

Synopsis

```
net <ads|rap|rpc> [-h] [-w workgroup] [-W myworkgroup] [-U user] [-I  
    ip-address] [-p port] [-n myname] [-s conffile] [-S server] [-l] [-P]  
    [-D debuglevel]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

The samba net utility is meant to work just like the net utility available for windows and DOS. The first argument should be used to specify the protocol to use when executing a certain command. ADS is used for ActiveDirectory, RAP is using for old (Win9x/NT3) clients and RPC can be used for NT4 and Windows 2000. If this argument is omitted, net will try to determine it automatically. Not all commands are available on all protocols.

OPTIONS

- h|--help** — Print a summary of command line options.

- w target-workgroup** — Sets target workgroup or domain. You have to specify either this option or the IP address or the name of a server.

- W workgroup** — Sets client workgroup or domain

- U user** — User name to use

- I ip-address** — IP address of target server to use. You have to specify either this option or a target workgroup or a target server.

- p port** — Port on the target server to connect to (usually 139 or 445). Defaults to trying 445 first, then 139.

- n <primary NetBIOS name>** — This option allows you to override the NetBIOS name that Samba uses for itself. This is identical to setting the *netbios name* parameter in the `smb.conf` file. However, a command line setting will take precedence over settings in `smb.conf`.

- s <configuration file>** — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

- S server** — Name of target server. You should specify either this option or a target workgroup or a target IP address.

- l** — When listing data, give more information on each item.

- P** — Make queries to the external server using the machine account of the local server.

-d|--debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

COMMANDS

CHANGESECRETPW

This command allows the Samba machine account password to be set from an external application to a machine account password that has already been stored in Active Directory. **DO NOT USE** this command unless you know exactly what you are doing. The use of this command requires that the force flag (-f) be used also. There will be **NO** command prompt. Whatever information is piped into stdin, either by typing at the command line or otherwise, will be stored as the literal machine password. Do **NOT** use this without care and attention as it will overwrite a legitimate machine password without warning. **YOU HAVE BEEN WARNED.**

TIME

The **NET TIME** command allows you to view the time on a remote server or synchronise the time on the local server with the time on the remote server.

TIME Without any options, the **NET TIME** command displays the time on the remote server.

TIME SYSTEM Displays the time on the remote server in a format ready for `/bin/date`

TIME SET Tries to set the date and time of the local server to that on the remote server using `/bin/date`.

TIME ZONE Displays the timezone in hours from GMT on the remote computer.

[RPC|ADS] JOIN [TYPE] [-U username[%password]] [options]

Join a domain. If the account already exists on the server, and [TYPE] is MEMBER, the machine will attempt to join automatically. (Assuming that the machine has been created in server manager) Otherwise, a password will be prompted for, and a new account may be created.

[TYPE] may be PDC, BDC or MEMBER to specify the type of server joining the domain.

[RPC] OLDJOIN [options]

Join a domain. Use the OLDJOIN option to join the domain using the old style of domain joining - you need to create a trust account in server manager first.

[RPC|ADS] USER

[RPC|ADS] USER DELETE *target* Delete specified user

[RPC|ADS] USER LIST List all users

[RPC|ADS] USER INFO *target* List the domain groups of a the specified user.

[RPC|ADS] USER ADD *name* [password] [-F user flags] [-C comment] Add specified user.

[RPC|ADS] GROUP

[RPC|ADS] GROUP [misc options] [targets] List user groups.

[RPC|ADS] GROUP DELETE *name* [misc. options] Delete specified group.

[RPC|ADS] GROUP ADD *name* [-C comment] Create specified group.

[RAP|RPC] SHARE

[RAP|RPC] SHARE [misc. options] [targets] Enumerates all exported resources (network shares) on target server.

[RAP|RPC] SHARE ADD *name=serverpath* [-C comment] [-M maxusers] [targets] Adds a share from a server (makes the export active). Maxusers specifies the number of users that can be connected to the share simultaneously.

SHARE DELETE *sharenam* Delete specified share.

[RPC|RAP] FILE

[RPC|RAP] FILE List all open files on remote server.

[RPC|RAP] FILE CLOSE *fileid* Close file with specified *fileid* on remote server.

[RPC|RAP] FILE INFO *fileid* Print information on specified *fileid*. Currently listed are: file-id, username, locks, path, permissions.

[RAP|RPC] FILE USER

NOTE



Currently NOT implemented.

SESSION

RAP SESSION Without any other options, SESSION enumerates all active SMB/CIFS sessions on the target server.

RAP SESSION DELETE|CLOSE *CLIENT_NAME* Close the specified sessions.

RAP SESSION INFO *CLIENT_NAME* Give a list with all the open files in specified session.

RAP SERVER *DOMAIN*

List all servers in specified domain or workgroup. Defaults to local domain.

RAP DOMAIN

Lists all domains and workgroups visible on the current network.

RAP PRINTQ

RAP PRINTQ LIST *QUEUE_NAME* Lists the specified print queue and print jobs on the server. If the *QUEUE_NAME* is omitted, all queues are listed.

RAP PRINTQ DELETE *JOBID* Delete job with specified id.

RAP VALIDATE *user* [*password*]

Validate whether the specified user can log in to the remote server. If the password is not specified on the commandline, it will be prompted.

NOTE



Currently NOT implemented.

RAP GROUPMEMBER

RAP GROUPMEMBER LIST *GROUP* List all members of the specified group.

RAP GROUPMEMBER DELETE *GROUP USER* Delete member from group.

RAP GROUPMEMBER ADD *GROUP USER* Add member to group.

RAP ADMIN *command*

Execute the specified *command* on the remote server. Only works with OS/2 servers.

NOTE



Currently NOT implemented.

RAP SERVICE

RAP SERVICE START *NAME* [**arguments...**] Start the specified service on the remote server. Not implemented yet.

NOTE



Currently NOT implemented.

RAP SERVICE STOP Stop the specified service on the remote server.

NOTE



Currently NOT implemented.

RAP PASSWORD *USER OLDPASS NEWPASS*

Change password of *USER* from *OLDPASS* to *NEWPASS*.

LOOKUP

LOOKUP HOST *HOSTNAME* [*TYPE*] Lookup the IP address of the given host with the specified type (netbios suffix). The type defaults to 0x20 (workstation).

LOOKUP LDAP [*DOMAIN*] Give IP address of LDAP server of specified *DOMAIN*. Defaults to local domain.

LOOKUP KDC [*REALM*] Give IP address of KDC for the specified *REALM*. Defaults to local realm.

LOOKUP DC [*DOMAIN*] Give IP's of Domain Controllers for specified *DOMAIN*. Defaults to local domain.

LOOKUP MASTER *DOMAIN* Give IP of master browser for specified *DOMAIN* or workgroup. Defaults to local domain.

CACHE

Samba uses a general caching interface called 'gencache'. It can be controlled using 'NET CACHE'.

s - Seconds

m - Minutes

All the timeout parameters support the suffixes: h - Hours

d - Days

w - Weeks

CACHE ADD *key data time-out* Add specified key+data to the cache with the given timeout.

CACHE DEL *key* Delete key from the cache.

CACHE SET *key data time-out* Update data of existing cache entry.

CACHE SEARCH *PATTERN* Search for the specified pattern in the cache data.

CACHE LIST List all current items in the cache.

CACHE FLUSH Remove all the current items from the cache.

GETLOCALSID [DOMAIN]

Print the SID of the specified domain, or if the parameter is omitted, the SID of the domain the local server is in.

SETLOCALSID S-1-5-21-x-y-z

Sets domain sid for the local server to the specified SID.

GROUPMAP

Manage the mappings between Windows group SIDs and UNIX groups. Parameters take the form "parameter=value". Common options include:

- **unixgroup** - Name of the UNIX group
- **ntgroup** - Name of the Windows NT group (must be resolvable to a SID)
- **rid** - Unsigned 32-bit integer
- **sid** - Full SID in the form of "S-1-..."
- **type** - Type of the group; either 'domain', 'local', or 'builtin'
- **comment** - Freeform text description of the group

GROUPMAP ADD Add a new group mapping entry

```
net groupmap add {rid=int|sid=string} unixgroup=string [type={domain|local|builtin}]
[ntgroup=string] [comment=string]
```

GROUPMAP DELETE Delete a group mapping entry

```
net groupmap delete {ntgroup=string|sid=SID}
```

GROUPMAP MODIFY Update an existing group entry

```
net groupmap modify {ntgroup=string|sid=SID}
[unixgroup=string] [comment=string] [type={domain|local}]
```

GROUPMAP LIST List existing group mapping entries

```
net groupmap list [verbose] [ntgroup=string] [sid=SID]
```


MAXRID

Prints out the highest RID currently in use on the local server (by the active 'passdb backend').

RPC INFO

Print information about the domain of the remote server, such as domain name, domain sid and number of users and groups.

[RPC|ADS] TESTJOIN

Check whether participation in a domain is still valid.

[RPC|ADS] CHANGETRUSTPW

Force change of domain trust password.

RPC TRUSTDOM

RPC TRUSTDOM ADD *DOMAIN* Add a interdomain trust account for *DOMAIN* to the remote server.

RPC TRUSTDOM DEL *DOMAIN* Remove interdomain trust account for *DOMAIN* from the remote server.

NOTE



Currently NOT implemented.

RPC TRUSTDOM ESTABLISH *DOMAIN* Establish a trust relationship to a trusting domain. Interdomain account must already be created on the remote PDC.

RPC TRUSTDOM REVOKE *DOMAIN* Abandon relationship to trusted domain

RPC TRUSTDOM LIST List all current interdomain trust relationships.

RPC ABORTSHUTDOWN

Abort the shutdown of a remote server.

SHUTDOWN [-t timeout] [-r] [-f] [-C message]

Shut down the remote server.

-r — Reboot after shutdown.

-f — Force shutting down all applications.

-t timeout — Timeout before system will be shut down. An interactive user of the system can use this time to cancel the shutdown.

-C message — Display the specified message on the screen to announce the shutdown.

SAMDUMP

Print out sam database of remote server. You need to run this on either a BDC.

VAMPIRE

Export users, aliases and groups from remote server to local server. Can only be run on a BDC.

GETSID

Fetch domain SID and store it in the local `secrets.tdb`.

ADS LEAVE

Make the remote host leave the domain it is part of.

ADS STATUS

Print out status of machine account of the local machine in ADS. Prints out quite some debug info. Aimed at developers, regular users should use **NET ADS TESTJOIN**.

ADS PRINTER

ADS PRINTER INFO [*PRINTER*] [*SERVER*] Lookup info for *PRINTER* on *SERVER*. The printer name defaults to `""`, the server name defaults to the local host.

ADS PRINTER PUBLISH *PRINTER* Publish specified printer using ADS.

ADS PRINTER REMOVE *PRINTER* Remove specified printer from ADS directory.

ADS SEARCH *EXPRESSION ATTRIBUTES...*

Perform a raw LDAP search on a ADS server and dump the results. The expression is a standard LDAP search expression, and the attributes are a list of LDAP fields to show in the results.

Example: `net ads search '(objectCategory=group)' sAMAccountName`

ADS DN *DN (attributes)*

Perform a raw LDAP search on a ADS server and dump the results. The DN standard LDAP DN, and the attributes are a list of LDAP fields to show in the result.

Example: `net ads dn 'CN=administrator,CN=Users,DC=my,DC=domain' SAMAccountName`

WORKGROUP

Print out workgroup name for specified kerberos realm.

HELP [COMMAND]

Gives usage information for the specified command.

A.7 nmbd**Synopsis**

```
nmbd [-D] [-F] [-S] [-a] [-i] [-o] [-h] [-V] [-d <debug level>] [-H
    <lmhosts file>] [-l <log directory>] [-n <primary netbios name>] [-p
    <port number>] [-s <configuration file>]
```

DESCRIPTION

This program is part of the Samba(7) suite.

nmbd is a server that understands and can reply to NetBIOS over IP name service requests, like those produced by SMB/CIFS clients such as Windows 95/98/Me, Windows NT, Windows 2000, Windows XP and LanManager clients. It also participates in the browsing protocols which make up the Windows "Network Neighborhood" view.

SMB/CIFS clients, when they start up, may wish to locate an SMB/CIFS server. That is, they wish to know what IP number a specified host is using.

Among other services, **nmbd** will listen for such requests, and if its own NetBIOS name is specified it will respond with the IP number of the host it is running on. Its "own NetBIOS name" is by default the primary DNS name of the host it is running on, but this can be overridden with the *-n* option (see OPTIONS below). Thus **nmbd** will reply to broadcast

queries for its own name(s). Additional names for **nmbd** to respond on can be set via parameters in the `smb.conf(5)` configuration file.

nmbd can also be used as a WINS (Windows Internet Name Server) server. What this basically means is that it will act as a WINS database server, creating a database from name registration requests that it receives and replying to queries from clients for these names.

In addition, **nmbd** can act as a WINS proxy, relaying broadcast queries from clients that do not understand how to talk the WINS protocol to a WINS server.

OPTIONS

- D** — If specified, this parameter causes **nmbd** to operate as a daemon. That is, it detaches itself and runs in the background, fielding requests on the appropriate port. By default, **nmbd** will operate as a daemon if launched from a command shell. **nmbd** can also be operated from the **inetd** meta-daemon, although this is not recommended.
- F** — If specified, this parameter causes the main **nmbd** process to not daemonize, i.e. double-fork and disassociate with the terminal. Child processes are still created as normal to service each connection request, but the main process does not exit. This operation mode is suitable for running **nmbd** under process supervisors such as **supervise** and **svscan** from Daniel J. Bernstein's **daemontools** package, or the AIX process monitor.
- S** — If specified, this parameter causes **nmbd** to log to standard output rather than a file.
- i** — If this parameter is specified it causes the server to run "interactively", not as a daemon, even if the server is executed on the command line of a shell. Setting this parameter negates the implicit daemon mode when run from the command line. **nmbd** also logs to standard output, as if the **-S** parameter had been given.
- h|–help** — Print a summary of command line options.
- H <filename>** — NetBIOS lmhosts file. The lmhosts file is a list of NetBIOS names to IP addresses that is loaded by the **nmbd** server and used via the name resolution mechanism *name resolve order* described in `smb.conf(5)` to resolve any NetBIOS name queries needed by the server. Note that the contents of this file are *NOT* used by **nmbd** to answer any name queries. Adding a line to this file affects name NetBIOS resolution from this host *ONLY*.

The default path to this file is compiled into Samba as part of the build process. Common defaults are `/usr/local/samba/lib/lmhosts`, `/usr/samba/lib/lmhosts` or `/etc/samba/lmhosts`. See the `lmhosts(5)` man page for details on the contents of this file.

-V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|—debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|—logfile=logbasename — File name for log/debug files. The extension ".client" will be appended. The log file is never removed by the client.

-p <UDP port number> — UDP port number is a positive integer value. This option changes the default UDP port number (normally 137) that **nmbd** responds to name queries on. Don't use this option unless you are an expert, in which case you won't need help!

FILES

`/etc/inetd.conf` — If the server is to be run by the **inetd** meta-daemon, this file must contain suitable startup information for the meta-daemon.

`/etc/rc` — or whatever initialization script your system uses.

If running the server as a daemon at startup, this file will need to contain an appropriate startup sequence for the server.

`/etc/services` — If running the server via the meta-daemon **inetd**, this file must contain a mapping of service name (e.g., netbios-ssn) to service port (e.g., 139) and protocol type (e.g., tcp).

`/usr/local/samba/lib/smb.conf` — This is the default location of the `smb.conf(5)` server configuration file. Other common places that systems install this file are `/usr/samba/lib/smb.conf` and `/etc/samba/smb.conf`.

When run as a WINS server (see the *wins support* parameter in the `smb.conf(5)` man page), **nmbd** will store the WINS database in the file `wins.dat` in the `var/locks` directory configured under wherever Samba was configured to install itself.

If **nmbd** is acting as a *browse master* (see the *local master* parameter in the `smb.conf(5)` man page, **nmbd** will store the browsing database in the file `browse.dat` in the `var/locks` directory configured under wherever Samba was configured to install itself.

SIGNALS

To shut down an **nmbd** process it is recommended that SIGKILL (-9) *NOT* be used, except as a last resort, as this may leave the name database in an inconsistent state. The correct way to terminate **nmbd** is to send it a SIGTERM (-15) signal and wait for it to die on its own.

nmbd will accept SIGHUP, which will cause it to dump out its namelists into the file `namelist.debug` in the `/usr/local/samba/var/locks` directory (or the `var/locks` directory configured under wherever Samba was configured to install itself). This will also cause **nmbd** to dump out its server database in the `log.nmb` file.

The debug log level of `nmbd` may be raised or lowered using `smbcontrol(1)` (SIGUSR[1|2] signals are no longer used since Samba 2.2). This is to allow transient problems to be diagnosed, whilst still running at a normally low log level.

SEE ALSO

`inetd(8)`, `smbd(8)`, `smb.conf(5)`, `smbclient(1)`, `testparm(1)`, `testprns(1)`, and the Internet RFC's `rfc1001.txt`, `rfc1002.txt`. In addition the CIFS (formerly SMB) specification is available as a link from the Web page <http://samba.org/cifs/>.

A.8 pdbedit

Synopsis

```

pdbedit [-L] [-v] [-w] [-u username] [-f fullname] [-h homedir] [-D drive]
        [-S script] [-p profile] [-a] [-m] [-r] [-x] [-i passdb-backend] [-e
        passdb-backend] [-b passdb-backend] [-g] [-d debuglevel] [-s
        configfile] [-P account-policy] [-C value] [-c account-control]

```

DESCRIPTION

This tool is part of the Samba(7) suite.

The pdbedit program is used to manage the users accounts stored in the sam database and can only be run by root.

The pdbedit tool uses the passdb modular interface and is independent from the kind of users database used (currently there are smbpasswd, ldap, nis+ and tdb based and more can be added without changing the tool).

There are five main ways to use pdbedit: adding a user account, removing a user account, modifying a user account, listing user accounts, importing users accounts.

OPTIONS

-L — This option lists all the user accounts present in the users database. This option prints a list of user/uid pairs separated by the ':' character.

Example: **pdbedit -L**

```
sorce:500:Simo Sorce
samba:45:Test User
```

-v — This option enables the verbose listing format. It causes pdbedit to list the users in the database, printing out the account fields in a descriptive format.

Example: **pdbedit -L -v**

```
-----
username:      sorce
user ID/Group: 500/500
user RID/GRID: 2000/2001
Full Name:     Simo Sorce
Home Directory: \\BERSERKER\sorce
HomeDir Drive: H:
Logon Script:  \\BERSERKER\netlogon\sorce.bat
Profile Path:  \\BERSERKER\profile
-----
username:      samba
user ID/Group: 45/45
user RID/GRID: 1090/1091
Full Name:     Test User
Home Directory: \\BERSERKER\samba
HomeDir Drive:
Logon Script:
```

Profile Path: \\BERSERKER\profile

-w — This option sets the "smbpasswd" listing format. It will make pdbedit list the users in the database, printing out the account fields in a format compatible with the smbpasswd file format. (see the smbpasswd(5) for details)

Example: **pdbedit -L -w**

```
sorce:500:508818B733CE64BEAAD3B435B51404EE \
      :D2A2418EFC466A8A0F6B1DBB5C3DB80C \
      :[UX           ]:LCT-00000000:
samba:45:0F2B255F7B67A7A9AAD3B435B51404EE \
      :BC281CE3F53B6A5146629CD4751D3490 \
      :[UX           ]:LCT-3BFA1E8D:
```

-u username — This option specifies the username to be used for the operation requested (listing, adding, removing). It is *required* in add, remove and modify operations and *optional* in list operations.

-f fullname — This option can be used while adding or modifying a user account. It will specify the user's full name.

Example: **-f "Simo Sorce"**

-h homedir — This option can be used while adding or modifying a user account. It will specify the user's home directory network path.

Example: **-h "\\BERSERKER\sorce"**

-D drive — This option can be used while adding or modifying a user account. It will specify the windows drive letter to be used to map the home directory.

Example: **-d "H:"**

-S script — This option can be used while adding or modifying a user account. It will specify the user's logon script path.

Example: **-s "\\BERSERKER\netlogon\sorce.bat"**

-p profile — This option can be used while adding or modifying a user account. It will specify the user's profile directory.

Example: **-p "\\BERSERKER\netlogon"**

-G SID|rid — This option can be used while adding or modifying a user account. It will specify the users' new primary group SID (Security Identifier) or rid.

Example: **-G S-1-5-21-2447931902-1787058256-3961074038-1201**

-U SID|rid — This option can be used while adding or modifying a user account. It will specify the users' new SID (Security Identifier) or rid.

Example: **-U S-1-5-21-2447931902-1787058256-3961074038-5004**

-c account-control — This option can be used while adding or modifying a user account. It will specify the users' account control property. Possible flags that can be set are: N, D, H, L, X.

Example: **-c "[X]"**

-a — This option is used to add a user into the database. This command needs a user name specified with the `-u` switch. When adding a new user, `pdbedit` will also ask for the password to be used.

Example: **pdbedit -a -u sorce**

```
new password:  
retype new password
```

-r — This option is used to modify an existing user in the database. This command needs a user name specified with the `-u` switch. Other options can be specified to modify the properties of the specified user. This flag is kept for backwards compatibility, but it is no longer necessary to specify it.

-m — This option may only be used in conjunction with the `-a` option. It will make `pdbedit` to add a machine trust account instead of a user account (`-u` username will provide the machine name).

Example: **pdbedit -a -m -u w2k-wks**

-x — This option causes `pdbedit` to delete an account from the database. It needs a username specified with the `-u` switch.

Example: **pdbedit -x -u bob**

-i passdb-backend — Use a different `passdb` backend to retrieve users than the one specified in `smb.conf`. Can be used to import data into your local user database.

This option will ease migration from one `passdb` backend to another.

Example: **pdbedit -i smbpasswd:/etc/smbpasswd.old**

-e passwd-backend — Exports all currently available users to the specified password database backend.

This option will ease migration from one passwd backend to another and will ease backing up.

Example: `pdbedit -e smbpasswd:/root/samba-users.backup`

-g — If you specify **-g**, then **-i in-backend -e out-backend** applies to the group mapping instead of the user database.

This option will ease migration from one passwd backend to another and will ease backing up.

-b passwd-backend — Use a different default passwd backend.

Example: `pdbedit -b xml:/root/pdb-backup.xml -l`

-P account-policy — Display an account policy

Valid policies are: minimum password age, reset count minutes, disconnect time, user must logon to change password, password history, lockout duration, min password length, maximum password age and bad lockout attempt.

Example: `pdbedit -P "bad lockout attempt"`

```
account policy value for bad lockout attempt is 0
```

-C account-policy-value — Sets an account policy to a specified value. This option may only be used in conjunction with the **-P** option.

Example: `pdbedit -P "bad lockout attempt" -C 3`

```
account policy value for bad lockout attempt was 0
account policy value for bad lockout attempt is now 3
```

-h|--help — Print a summary of command line options.

-V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is

to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|--debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|--logfile=logbasename — File name for log/debug files. The extension `.client` will be appended. The log file is never removed by the client.

NOTES

This command may be used only by root.

SEE ALSO

`smbpasswd(5)`, `samba(7)`

A.9 `smbcquotas`

Synopsis

```
smbcquotas //server/share [-u user] [-L] [-F] [-S QUOTA_SET_COMMAND] [-n]
    [-t] [-v] [-d debuglevel] [-s configfile] [-l logfilebase] [-V] [-U
    username] [-N] [-k] [-A]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

The `smbcquotas` program manipulates NT Quotas on SMB file shares.

OPTIONS

The following options are available to the **smbquotas** program.

- u user** — Specifies the user of whom the quotas are get or set. By default the current user's username will be used.

- L** — Lists all quota records of the share.

- F** — Show the share quota status and default limits.

- S QUOTA_SET_COMMAND** — This command sets/modifies quotas for a user or on the share, depending on the QUOTA_SET_COMMAND parameter which is described later.

- n** — This option displays all QUOTA information in numeric format. The default is to convert SIDs to names and QUOTA limits to a readable string format.

- t** — Don't actually do anything, only validate the correctness of the arguments.

- v** — Be verbose.

- h|—help** — Print a summary of command line options.

- V** — Prints the program version number.

- s <configuration file>** — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See **smb.conf** for more information. The default configuration file name is determined at compile time.

- d|—debug=debuglevel** — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|-logfile=logbasename — File name for log/debug files. The extension ".client" will be appended. The log file is never removed by the client.

-N — If specified, this parameter suppresses the normal password prompt from the client to the user. This is useful when accessing a service that does not require a password.

Unless a password is specified on the command line or this parameter is specified, the client will request a password.

-k — Try to authenticate with kerberos. Only useful in an Active Directory environment.

-A|-authfile=filename — This option allows you to specify a file from which to read the username and password used in the connection. The format of the file is

```
username = <value>
password = <value>
domain   = <value>
```

Make certain that the permissions on the file restrict access from unwanted users.

-U|-user=username[%password —] Sets the SMB username or username and password.

If %password is not specified, the user will be prompted. The client will first check the `USER` environment variable, then the `LOGNAME` variable and if either exists, the string is uppercased. If these environmental variables are not found, the username `GUEST` is used.

A third option is to use a credentials file which contains the plaintext of the username and password. This option is mainly provided for scripts where the admin does not wish to pass the credentials on the command line or via environment variables. If this method is used, make certain that the permissions on the file restrict access from unwanted users. See the **-A** for more details.

Be cautious about including passwords in scripts. Also, on many systems the command line of a running process may be seen via the `ps` command. To be safe always allow `rpcclient` to prompt for a password and type it in directly.

QUOTA_SET_COMAND

The format of an ACL is one or more ACL entries separated by either commas or newlines. An ACL entry is one of the following:

for setting user quotas for the user specified by `-u` or the current username:

```
UQLIM:<username>:<softlimit>/<hardlimit>
```

for setting the default quotas for a share:

```
FSQLIM:<softlimit>/<hardlimit>
```

for changing the share quota settings:

```
FSQFLAGS:QUOTA_ENABLED/DENY_DISK/LOG_SOFTLIMIT/LOG_HARD_LIMIT
```

EXIT STATUS

The **smbcquotas** program sets the exit status depending on the success or otherwise of the operations performed. The exit status may be one of the following values.

If the operation succeeded, **smbcquotas** returns an exit status of 0. If **smbcquotas** couldn't connect to the specified server, or when there was an error getting or setting the quota(s), an exit status of 1 is returned. If there was an error parsing any command line arguments, an exit status of 2 is returned.

A.10 **smbd**

Synopsis

```
smbd [-D] [-F] [-S] [-i] [-h] [-V] [-b] [-d <debug level>] [-l <log
    directory>] [-p <port number>] [-O <socket option>] [-s
    <configuration file>]
```

DESCRIPTION

This program is part of the Samba(7) suite.

smbd is the server daemon that provides filesharing and printing services to Windows clients. The server provides filespace and printer services to clients using the SMB (or CIFS) protocol. This is compatible with the LanManager protocol, and can service LanManager clients. These include MSCLIENT 3.0 for DOS, Windows for Workgroups, Windows 95/98/ME, Windows NT, Windows 2000, OS/2, DAVE for Macintosh, and smbfs for Linux.

An extensive description of the services that the server can provide is given in the man page for the configuration file controlling the attributes of those services (see `smb.conf(5)`). This man page will not describe the services, but will concentrate on the administrative aspects of running the server.

Please note that there are significant security implications to running this server, and the `smb.conf(5)` manual page should be regarded as mandatory reading before proceeding with installation.

A session is created whenever a client requests one. Each client gets a copy of the server for each session. This copy then services all connections made by the client during that session. When all connections from its client are closed, the copy of the server for that client terminates.

The configuration file, and any files that it includes, are automatically reloaded every minute, if they change. You can force a reload by sending a `SIGHUP` to the server. Reloading the configuration file will not affect connections to any service that is already established. Either the user will have to disconnect from the service, or `smbd` killed and restarted.

OPTIONS

- D** — If specified, this parameter causes the server to operate as a daemon. That is, it detaches itself and runs in the background, fielding requests on the appropriate port. Operating the server as a daemon is the recommended way of running `smbd` for servers that provide more than casual use file and print services. This switch is assumed if `smbd` is executed on the command line of a shell.

- F** — If specified, this parameter causes the main `smbd` process to not daemonize, i.e. double-fork and disassociate with the terminal. Child processes are still created as normal to service each connection request, but the main process does not exit. This operation mode is suitable for running `smbd` under process supervisors such as `supervise` and `svscan` from Daniel J. Bernstein's `daemontools` package, or the AIX process monitor.

- S** — If specified, this parameter causes `smbd` to log to standard output rather than a file.

- i** — If this parameter is specified it causes the server to run "interactively", not as a daemon, even if the server is executed on the command line of a shell. Setting this parameter negates the implicit daemon mode when run from the command line. `smbd` also logs to standard output, as if the **-S** parameter had been given.

- V** — Prints the program version number.

- s <configuration file>** — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

- d|--debug=debuglevel** — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used

when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|--logfile=logbasename — File name for log/debug files. The extension ".client" will be appended. The log file is never removed by the client.

-h|--help — Print a summary of command line options.

-b — Prints information about how Samba was built.

-l <log directory> — If specified, *log directory* specifies a log directory into which the "log.smbd" log file will be created for informational and debug messages from the running server. The log file generated is never removed by the server although its size may be controlled by the *max log size* option in the `smb.conf(5)` file. *Beware:* If the directory specified does not exist, **smbd** will log to the default debug log location defined at compile time.

The default log directory is specified at compile time.

-p <port number> — *port number* is a positive integer value. The default value if this parameter is not specified is 139.

This number is the port number that will be used when making connections to the server from client software. The standard (well-known) port number for the SMB over TCP is 139, hence the default. If you wish to run the server as an ordinary user rather than as root, most systems will require you to use a port number greater than 1024 - ask your system administrator for help if you are in this situation.

In order for the server to be useful by most clients, should you configure it on a port other than 139, you will require port redirection services on port 139, details of which are outlined in `rfc1002.txt` section 4.3.5.

This parameter is not normally specified except in the above situation.

FILES

`/etc/inetd.conf` — If the server is to be run by the **inetd** meta-daemon, this file must contain suitable startup information for the meta-daemon.

`/etc/rc` — or whatever initialization script your system uses).

If running the server as a daemon at startup, this file will need to contain an appropriate startup sequence for the server.

`/etc/services` — If running the server via the meta-daemon **inetd**, this file must contain a mapping of service name (e.g., `netbios-ssn`) to service port (e.g., 139) and protocol type (e.g., `tcp`).

`/usr/local/samba/lib/smb.conf` — This is the default location of the `smb.conf(5)` server configuration file. Other common places that systems install this file are `/usr/samba/lib/smb.conf` and `/etc/samba/smb.conf`.

This file describes all the services the server is to make available to clients. See `smb.conf(5)` for more information.

LIMITATIONS

On some systems **smbd** cannot change uid back to root after a `setuid()` call. Such systems are called trapdoor uid systems. If you have such a system, you will be unable to connect from a client (such as a PC) as two different users at once. Attempts to connect the second user will result in access denied or similar.

ENVIRONMENT VARIABLES

PRINTER — If no printer name is specified to printable services, most systems will use the value of this variable (or `lp` if this variable is not defined) as the name of the printer to use. This is not specific to the server, however.

PAM INTERACTION

Samba uses PAM for authentication (when presented with a plaintext password), for account checking (is this account disabled?) and for session management. The degree to which samba supports PAM is restricted by the limitations of the SMB protocol and the *obey pam restrictions* `smb.conf(5)` parameter. When this is set, the following restrictions apply:

- *Account Validation*: All accesses to a samba server are checked against PAM to see if the account is valid, not disabled and is permitted to login at this time. This also applies to encrypted logins.

- *Session Management:* When not using share level security, users must pass PAM's session checks before access is granted. Note however, that this is bypassed in share level security. Note also that some older pam configuration files may need a line added for session support.

DIAGNOSTICS

Most diagnostics issued by the server are logged in a specified log file. The log file name is specified at compile time, but may be overridden on the command line.

The number and nature of diagnostics available depends on the debug level used by the server. If you have problems, set the debug level to 3 and peruse the log files.

Most messages are reasonably self-explanatory. Unfortunately, at the time this man page was created, there are too many diagnostics available in the source code to warrant describing each and every diagnostic. At this stage your best bet is still to grep the source code and inspect the conditions that gave rise to the diagnostics you are seeing.

SIGNALS

Sending the **smbd** a SIGHUP will cause it to reload its **smb.conf** configuration file within a short period of time.

To shut down a user's **smbd** process it is recommended that **SIGKILL (-9)** *NOT* be used, except as a last resort, as this may leave the shared memory area in an inconsistent state. The safe way to terminate an **smbd** is to send it a SIGTERM (-15) signal and wait for it to die on its own.

The debug log level of **smbd** may be raised or lowered using **smbcontrol(1)** program (SIGUSR[1|2] signals are no longer used since Samba 2.2). This is to allow transient problems to be diagnosed, whilst still running at a normally low log level.

Note that as the signal handlers send a debug write, they are not re-entrant in **smbd**. This you should wait until **smbd** is in a state of waiting for an incoming SMB before issuing them. It is possible to make the signal handlers safe by un-blocking the signals before the select call and re-blocking them after, however this would affect performance.

SEE ALSO

hosts_access(5), **inetd(8)**, **nmbd(8)**, **smb.conf(5)**, **smbclient(1)**, **testparm(1)**, **testprns(1)**, and the Internet RFC's **rfc1001.txt**, **rfc1002.txt**. In addition the CIFS (formerly SMB) specification is available as a link from the Web page <http://samba.org/cifs/>.

A.11 smbpasswd

Synopsis

smbpasswd

DESCRIPTION

This tool is part of the Samba(7) suite.

smbpasswd is the Samba encrypted password file. It contains the username, Unix user id and the SMB hashed passwords of the user, as well as account flag information and the time the password was last changed. This file format has been evolving with Samba and has had several different formats in the past.

FILE FORMAT

The format of the smbpasswd file used by Samba 2.2 is very similar to the familiar Unix `passwd(5)` file. It is an ASCII file containing one line for each user. Each field within each line is separated from the next by a colon. Any entry beginning with '#' is ignored. The smbpasswd file contains the following information for each user:

name — This is the user name. It must be a name that already exists in the standard UNIX `passwd` file.

uid — This is the UNIX uid. It must match the uid field for the same user entry in the standard UNIX `passwd` file. If this does not match then Samba will refuse to recognize this smbpasswd file entry as being valid for a user.

Lanman Password Hash — This is the LANMAN hash of the user's password, encoded as 32 hex digits. The LANMAN hash is created by DES encrypting a well known string with the user's password as the DES key. This is the same password used by Windows 95/98 machines. Note that this password hash is regarded as weak as it is vulnerable to dictionary attacks and if two users choose the same password this entry will be identical (i.e. the password is not "salted" as the UNIX password is). If the user has a null password this field will contain the characters "NO PASSWORD" as the start of the hex string. If the hex string is equal to 32 'X' characters then the user's account is marked as `disabled` and the user will not be able to log onto the Samba server.

WARNING! Due to the challenge-response nature of the SMB/CIFS authentication protocol, anyone with a knowledge of this password hash will be able to impersonate the user on the network. For this reason these hashes are known as *plain text equivalents* and must *NOT* be made available to anyone but the root user. To protect these passwords the smbpasswd file is placed in a directory with read and traverse access only to the root user and the smbpasswd file itself must be set to be read/write only by root, with no other access.

NT Password Hash — This is the Windows NT hash of the user's password, encoded as 32 hex digits. The Windows NT hash is created by taking the user's password as represented in 16-bit, little-endian UNICODE and then applying the MD4 (internet rfc1321) hashing algorithm to it.

This password hash is considered more secure than the LANMAN Password Hash as it preserves the case of the password and uses a much higher quality hashing algorithm. However, it is still the case that if two users choose the same password this entry will be identical (i.e. the password is not "salted" as the UNIX password is).

WARNING !!. Note that, due to the challenge-response nature of the SMB/CIFS authentication protocol, anyone with a knowledge of this password hash will be able to impersonate the user on the network. For this reason these hashes are known as *plain text equivalents* and must *NOT* be made available to anyone but the root user. To protect these passwords the smbpasswd file is placed in a directory with read and traverse access only to the root user and the smbpasswd file itself must be set to be read/write only by root, with no other access.

Account Flags — This section contains flags that describe the attributes of the users account. In the Samba 2.2 release this field is bracketed by '[' and ']' characters and is always 13 characters in length (including the '[' and ']' characters). The contents of this field may be any of the following characters:

- *U* - This means this is a "User" account, i.e. an ordinary user. Only User and Workstation Trust accounts are currently supported in the smbpasswd file.
- *N* - This means the account has no password (the passwords in the fields LANMAN Password Hash and NT Password Hash are ignored). Note that this will only allow users to log on with no password if the *null passwords* parameter is set in the smb.conf(5) config file.
- *D* - This means the account is disabled and no SMB/CIFS logins will be allowed for this user.
- *W* - This means this account is a "Workstation Trust" account. This kind of account is used in the Samba PDC code stream to allow Windows NT Workstations and Servers to join a Domain hosted by a Samba PDC.

Other flags may be added as the code is extended in future. The rest of this field space is filled in with spaces.

Last Change Time — This field consists of the time the account was last modified. It consists of the characters 'LCT-' (standing for "Last Change Time") followed by a numeric encoding of the UNIX time in seconds since the epoch (1970) that the last change was made.

All other colon separated fields are ignored at this time.

SEE ALSO

smbpasswd(8), Samba(7), and the Internet RFC1321 for details on the MD4 algorithm.

A.12 smbpasswd

Synopsis

```
smbpasswd [-a] [-x] [-d] [-e] [-D debuglevel] [-n] [-r <remote machine>]
          [-R <name resolve order>] [-m] [-U username[%password]] [-h] [-s] [-w
          pass] [-i] [-L] [username]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

The smbpasswd program has several different functions, depending on whether it is run by the *root* user or not. When run as a normal user it allows the user to change the password used for their SMB sessions on any machines that store SMB passwords.

By default (when run with no arguments) it will attempt to change the current user's SMB password on the local machine. This is similar to the way the **passwd(1)** program works. **smbpasswd** differs from how the passwd program works however in that it is not *setuid root* but works in a client-server mode and communicates with a locally running **smbd(8)**. As a consequence in order for this to succeed the **smbd** daemon must be running on the local machine. On a UNIX machine the encrypted SMB passwords are usually stored in the **smbpasswd(5)** file.

When run by an ordinary user with no options, smbpasswd will prompt them for their old SMB password and then ask them for their new password twice, to ensure that the new password was typed correctly. No passwords will be echoed on the screen whilst being typed. If you have a blank SMB password (specified by the string "NO PASSWORD" in the smbpasswd file) then just press the <Enter> key when asked for your old password.

smbpasswd can also be used by a normal user to change their SMB password on remote machines, such as Windows NT Primary Domain Controllers. See the (*-r*) and *-U* options below.

When run by root, smbpasswd allows new users to be added and deleted in the smbpasswd file, as well as allows changes to the attributes of the user in this file to be made. When run by root, **smbpasswd** accesses the local smbpasswd file directly, thus enabling changes to be made even if **smbd** is not running.

OPTIONS

-a — This option specifies that the username following should be added to the local smbpasswd file, with the new password typed (type <Enter> for the old password). This option is ignored if the username following already exists in the smbpasswd file and it is treated like a regular change password command. Note that the default **passwd** backends require the user to already exist in the system password file (usually **/etc/passwd**), else the request to add the user will fail.

This option is only available when running smbpasswd as root.

- x** — This option specifies that the username following should be deleted from the local `smbpasswd` file.

This option is only available when running `smbpasswd` as root.

- d** — This option specifies that the username following should be **disabled** in the local `smbpasswd` file. This is done by writing a 'D' flag into the account control space in the `smbpasswd` file. Once this is done all attempts to authenticate via SMB using this username will fail.

If the `smbpasswd` file is in the 'old' format (pre-Samba 2.0 format) there is no space in the user's password entry to write this information and the command will FAIL. See `smbpasswd(5)` for details on the 'old' and new password file formats.

This option is only available when running `smbpasswd` as root.

- e** — This option specifies that the username following should be **enabled** in the local `smbpasswd` file, if the account was previously disabled. If the account was not disabled this option has no effect. Once the account is enabled then the user will be able to authenticate via SMB once again.

If the `smbpasswd` file is in the 'old' format, then **smbpasswd** will FAIL to enable the account. See `smbpasswd(5)` for details on the 'old' and new password file formats.

This option is only available when running `smbpasswd` as root.

- D debuglevel** — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of `smbpasswd`. At level 0, only critical errors and serious warnings will be logged.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

- n** — This option specifies that the username following should have their password set to null (i.e. a blank password) in the local `smbpasswd` file. This is done by writing the string "NO PASSWORD" as the first part of the first password stored in the `smbpasswd` file.

Note that to allow users to logon to a Samba server once the password has been set to "NO PASSWORD" in the `smbpasswd` file the administrator must set the following parameter in the [global] section of the `smb.conf` file :

null passwords = yes

This option is only available when running `smbpasswd` as root.

-r remote machine name — This option allows a user to specify what machine they wish to change their password on. Without this parameter `smbpasswd` defaults to the local host. The *remote machine name* is the NetBIOS name of the SMB/CIFS server to contact to attempt the password change. This name is resolved into an IP address using the standard name resolution mechanism in all programs of the Samba suite. See the *-R name resolve order* parameter for details on changing this resolving mechanism.

The username whose password is changed is that of the current UNIX logged on user. See the *-U username* parameter for details on changing the password for a different username.

Note that if changing a Windows NT Domain password the remote machine specified must be the Primary Domain Controller for the domain (Backup Domain Controllers only have a read-only copy of the user account database and will not allow the password change).

Note that Windows 95/98 do not have a real password database so it is not possible to change passwords specifying a Win95/98 machine as remote machine target.

-R name resolve order — This option allows the user of `smbpasswd` to determine what name resolution services to use when looking up the NetBIOS name of the host being connected to.

The options are `:"lmhosts"`, `"host"`, `"wins"` and `"bcast"`. They cause names to be resolved as follows:

- **lmhosts**: Lookup an IP address in the Samba `lmhosts` file. If the line in `lmhosts` has no name type attached to the NetBIOS name (see the `lmhosts(5)` for details) then any name type matches for lookup.
- **host**: Do a standard host name to IP address resolution, using the system `/etc/hosts`, NIS, or DNS lookups. This method of name resolution is operating system depended for instance on IRIX or Solaris this may be controlled by the `/etc/nsswitch.conf` file). Note that this method is only used if the NetBIOS name type being queried is the `0x20` (server) name type, otherwise it is ignored.
- **wins**: Query a name with the IP address listed in the *wins server* parameter. If no WINS server has been specified this method will be ignored.
- **bcast**: Do a broadcast on each of the known local interfaces listed in the *interfaces* parameter. This is the least reliable of the name resolution methods as it depends on the target host being on a locally connected subnet.

The default order is **lmhosts**, **host**, **wins**, **bcast** and without this parameter or any entry in the `smb.conf(5)` file the name resolution methods will be attempted in this order.

-m — This option tells `smbpasswd` that the account being changed is a MACHINE account. Currently this is used when Samba is being used as an NT Primary Domain Controller.

This option is only available when running `smbpasswd` as root.

- U username** — This option may only be used in conjunction with the `-r` option. When changing a password on a remote machine it allows the user to specify the user name on that machine whose password will be changed. It is present to allow users who have different user names on different systems to change these passwords.

- h** — This option prints the help string for `smbpasswd`, selecting the correct one for running as root or as an ordinary user.

- s** — This option causes `smbpasswd` to be silent (i.e. not issue prompts) and to read its old and new passwords from standard input, rather than from `/dev/tty` (like the `passwd(1)` program does). This option is to aid people writing scripts to drive `smbpasswd`.

- w password** — This parameter is only available if Samba has been configured to use the experimental `—with-ldapsam` option. The `-w` switch is used to specify the password to be used with the `ldap admin dn`. Note that the password is stored in the `secrets.tdb` and is keyed off of the admin's DN. This means that if the value of `ldap admin dn` ever changes, the password will need to be manually updated as well.

- i** — This option tells `smbpasswd` that the account being changed is an interdomain trust account. Currently this is used when Samba is being used as an NT Primary Domain Controller. The account contains the info about another trusted domain.

This option is only available when running `smbpasswd` as root.

- L** — Run in local mode.

username — This specifies the username for all of the *root only* options to operate on. Only root can specify this parameter as only root has the permission needed to modify attributes directly in the local `smbpasswd` file.

NOTES

Since `smbpasswd` works in client-server mode communicating with a local `smbd` for a non-root user then the `smbd` daemon must be running for this to work. A common problem is to add a restriction to the hosts that may access the `smbd` running on the local machine by specifying either `allow hosts` or `deny hosts` entry in the `smb.conf(5)` file and neglecting to allow "localhost" access to the `smbd`.

In addition, the `smbpasswd` command is only useful if Samba has been set up to use encrypted passwords.

SEE ALSO

smbpasswd(5), Samba(7).

A.13 smbstatus

Synopsis

```
smbstatus [-P] [-b] [-d <debug level>] [-v] [-L] [-B] [-p] [-S] [-s  
    <configuration file>] [-u <username>]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

smbstatus is a very simple program to list the current Samba connections.

OPTIONS

-P|**-profile** — If samba has been compiled with the profiling option, print only the contents of the profiling shared memory area.

-b|**-brief** — gives brief output.

-V — Prints the program version number.

-s **<configuration file>** — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See **smb.conf** for more information. The default configuration file name is determined at compile time.

-d|**-debug=debuglevel** — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|--logfile=logbasename — File name for log/debug files. The extension `.client` will be appended. The log file is never removed by the client.

-v|--verbose — gives verbose output.

-L|--locks — causes `smbstatus` to only list locks.

-B|--byterange — causes `smbstatus` to include byte range locks.

-p|--processes — print a list of `smbd(8)` processes and exit. Useful for scripting.

-S|--shares — causes `smbstatus` to only list shares.

-h|--help — Print a summary of command line options.

-u|--user=<username> — selects information relevant to *username* only.

SEE ALSO

`smbd(8)` and `smb.conf(5)`.

A.14 smbtree

Synopsis

```
smbtree [-b] [-D] [-S]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

smbtree is a smb browser program in text mode. It is similar to the "Network Neighborhood" found on Windows computers. It prints a tree with all the known domains, the servers in those domains and the shares on the servers.

OPTIONS

- b — Query network nodes by sending requests as broadcasts instead of querying the (domain) master browser.
- D — Only print a list of all the domains known on broadcast or by the master browser
- S — Only print a list of all the domains and servers responding on broadcast or known by the master browser.
- V — Prints the program version number.

-s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|--debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|--logfile=logbasename — File name for log/debug files. The extension ".client" will be appended. The log file is never removed by the client.

-N — If specified, this parameter suppresses the normal password prompt from the client to the user. This is useful when accessing a service that does not require a password.

Unless a password is specified on the command line or this parameter is specified, the client will request a password.

-k — Try to authenticate with kerberos. Only useful in an Active Directory environment.

-A|--authfile=filename — This option allows you to specify a file from which to read the username and password used in the connection. The format of the file is

```
username = <value>
password = <value>
domain   = <value>
```

Make certain that the permissions on the file restrict access from unwanted users.

-U|-**user=username**[%**password** —] Sets the SMB username or username and password.

If %password is not specified, the user will be prompted. The client will first check the **USER** environment variable, then the **LOGNAME** variable and if either exists, the string is uppercased. If these environmental variables are not found, the username **GUEST** is used.

A third option is to use a credentials file which contains the plaintext of the username and password. This option is mainly provided for scripts where the admin does not wish to pass the credentials on the command line or via environment variables. If this method is used, make certain that the permissions on the file restrict access from unwanted users. See the **-A** for more details.

Be cautious about including passwords in scripts. Also, on many systems the command line of a running process may be seen via the **ps** command. To be safe always allow **rpcclient** to prompt for a password and type it in directly.

-h|-**help** — Print a summary of command line options.

A.15 testparm

Synopsis

```
testparm [-s] [-h] [-v] [-L <servername>] [-t <encoding>] config filename
        [hostname hostIP]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

testparm is a very simple test program to check an **smbd**(8) configuration file for internal correctness. If this program reports no problems, you can use the configuration file with confidence that **smbd** will successfully load the configuration file.

Note that this is *NOT* a guarantee that the services specified in the configuration file will be available or will operate as expected.

If the optional host name and host IP address are specified on the command line, this test program will run through the service entries reporting whether the specified host has access to each service.

If **testparm** finds an error in the **smb.conf** file it returns an exit code of 1 to the calling program, else it returns an exit code of 0. This allows shell scripts to test the output from **testparm**.

OPTIONS

-s — Without this option, **testparm** will prompt for a carriage return after printing the service names and before dumping the service definitions.

-h|--help — Print a summary of command line options.

-V — Prints the program version number.

-L servername — Sets the value of the %L macro to *servername*. This is useful for testing include files specified with the %L macro.

-v — If this option is specified, testparm will also output all options that were not used in **smb.conf(5)** and are thus set to their defaults.

-t encoding — Output data in specified encoding.

configfilename — This is the name of the configuration file to check. If this parameter is not present then the default **smb.conf(5)** file will be checked.

hostname — If this parameter and the following are specified, then **testparm** will examine the *hosts allow* and *hosts deny* parameters in the **smb.conf(5)** file to determine if the hostname with this IP address would be allowed access to the **smbd** server. If this parameter is supplied, the **hostIP** parameter must also be supplied.

hostIP — This is the IP address of the host specified in the previous parameter. This address must be supplied if the **hostname** parameter is supplied.

FILES

smb.conf(5) — This is usually the name of the configuration file used by **smbd(8)**.

DIAGNOSTICS

The program will issue a message saying whether the configuration file loaded OK or not. This message may be preceded by errors and warnings if the file did not load. If the file was loaded OK, the program then dumps all known service details to stdout.

SEE ALSO

smb.conf(5), smbld(8)

A.16 wbinfo

Synopsis

```
wbinfo [-u] [-g] [-N netbios-name] [-I ip] [-n name] [-s sid] [-U uid] [-G
gid] [-S sid] [-Y sid] [-t] [-m] [--sequence] [-r user] [-a
user%password] [--set-auth-user user%password] [--get-auth-user] [-p]
```

DESCRIPTION

This tool is part of the Samba(7) suite.

The **wbinfo** program queries and returns information created and used by the winbindd(8) daemon.

The winbindd(8) daemon must be configured and running for the **wbinfo** program to be able to return information.

OPTIONS

- u** — This option will list all users available in the Windows NT domain for which the winbindd(8) daemon is operating in. Users in all trusted domains will also be listed. Note that this operation does not assign user ids to any users that have not already been seen by winbindd(8) .
- g** — This option will list all groups available in the Windows NT domain for which the Samba(7) daemon is operating in. Groups in all trusted domains will also be listed. Note that this operation does not assign group ids to any groups that have not already been seen by winbindd(8).
- N name** — The *-N* option queries winbindd(8) to query the WINS server for the IP address associated with the NetBIOS name specified by the *name* parameter.
- I ip** — The *-I* option queries winbindd(8) to send a node status request to get the NetBIOS name associated with the IP address specified by the *ip* parameter.
- n name** — The *-n* option queries winbindd(8) for the SID associated with the name specified. Domain names can be specified before the user name by using the winbind

separator character. For example CWDOM1/Administrator refers to the Administrator user in the domain CWDOM1. If no domain is specified then the domain used is the one specified in the smb.conf(5) *workgroup* parameter.

- s sid** — Use *-s* to resolve a SID to a name. This is the inverse of the *-n* option above. SIDs must be specified as ASCII strings in the traditional Microsoft format. For example, S-1-5-21-1455342024-3071081365-2475485837-500.
- U uid** — Try to convert a UNIX user id to a Windows NT SID. If the uid specified does not refer to one within the idmap uid range then the operation will fail.
- G gid** — Try to convert a UNIX group id to a Windows NT SID. If the gid specified does not refer to one within the idmap gid range then the operation will fail.
- S sid** — Convert a SID to a UNIX user id. If the SID does not correspond to a UNIX user mapped by winbindd(8) then the operation will fail.
- Y sid** — Convert a SID to a UNIX group id. If the SID does not correspond to a UNIX group mapped by winbindd(8) then the operation will fail.
- t** — Verify that the workstation trust account created when the Samba server is added to the Windows NT domain is working.
- m** — Produce a list of domains trusted by the Windows NT server winbindd(8) contacts when resolving names. This list does not include the Windows NT domain the server is a Primary Domain Controller for.
- sequence** — Show sequence numbers of all known domains
- r username** — Try to obtain the list of UNIX group ids to which the user belongs. This only works for users defined on a Domain Controller.
- a username%password** — Attempt to authenticate a user via winbindd. This checks both authentication methods and reports its results.
- set-auth-user username%password** — Store username and password used by winbindd during session setup to a domain controller. This enables winbindd to operate in a Windows 2000 domain with Restrict Anonymous turned on (a.k.a. Permissions compatible with Windows 2000 servers only).
- get-auth-user** — Print username and password used by winbindd during session setup to a domain controller. Username and password can be set using *'-A'*. Only available

for root.

-p — Check whether winbindd is still alive. Prints out either 'succeeded' or 'failed'.

-V — Prints the program version number.

-h|–help — Print a summary of command line options.

EXIT STATUS

The wbinfo program returns 0 if the operation succeeded, or 1 if the operation failed. If the winbindd(8) daemon is not working **wbinfo** will always return failure.

SEE ALSO

winbindd(8)

A.17 winbindd

Synopsis

```
winbindd [-F] [-S] [-i] [-Y] [-d <debug level>] [-s <smb config file>]
        [-n]
```

DESCRIPTION

This program is part of the Samba(7) suite.

winbindd is a daemon that provides a service for the Name Service Switch capability that is present in most modern C libraries. The Name Service Switch allows user and system information to be obtained from different databases services such as NIS or DNS. The exact behaviour can be configured through the `/etc/nsswitch.conf` file. Users and groups are allocated as they are resolved to a range of user and group ids specified by the administrator of the Samba system.

The service provided by **winbindd** is called 'winbind' and can be used to resolve user and group information from a Windows NT server. The service can also provide authentication services via an associated PAM module.

The `pam.winbind` module in the 2.2.2 release only supports the *auth* and *account* module-types. The latter simply performs a `getpwnam()` to verify that the system can obtain a uid for the user. If the `libnss_winbind` library has been correctly installed, this should always succeed.

The following nsswitch databases are implemented by the winbindd service:

hosts — User information traditionally stored in the `hosts(5)` file and used by `gethostbyname(3)` functions. Names are resolved through the WINS server or by broadcast.

passwd — User information traditionally stored in the `passwd(5)` file and used by `getpwent(3)` functions.

group — Group information traditionally stored in the `group(5)` file and used by `getgrent(3)` functions.

For example, the following simple configuration in the `/etc/nsswitch.conf` file can be used to initially resolve user and group information from `/etc/passwd` and `/etc/group` and then from the Windows NT server.

```
passwd:      files winbind
group:      files winbind
```

The following simple configuration in the `/etc/nsswitch.conf` file can be used to initially resolve hostnames from `/etc/hosts` and then from the WINS server.

```
hosts:      files wins
```

OPTIONS

- F — If specified, this parameter causes the main **winbindd** process to not daemonize, i.e. double-fork and disassociate with the terminal. Child processes are still created as normal to service each connection request, but the main process does not exit. This operation mode is suitable for running **winbindd** under process supervisors such as **supervise** and **svscan** from Daniel J. Bernstein's **daemontools** package, or the AIX process monitor.
- S — If specified, this parameter causes **winbindd** to log to standard output rather than a file.
- V — Prints the program version number.
- s <configuration file> — The file specified contains the configuration details required by the server. The information in this file includes server-specific information such as what printcap file to use, as well as descriptions of all the services that the server is to provide. See `smb.conf` for more information. The default configuration file name is determined at compile time.

-d|—debug=debuglevel — *debuglevel* is an integer from 0 to 10. The default value if this parameter is not specified is zero.

The higher this value, the more detail will be logged to the log files about the activities of the server. At level 0, only critical errors and serious warnings will be logged. Level 1 is a reasonable level for day-to-day running - it generates a small amount of information about operations carried out.

Levels above 1 will generate considerable amounts of log data, and should only be used when investigating a problem. Levels above 3 are designed for use only by developers and generate HUGE amounts of log data, most of which is extremely cryptic.

Note that specifying this parameter here will override the *log level* parameter in the `smb.conf` file.

-l|—logfile=logbasename — File name for log/debug files. The extension ".client" will be appended. The log file is never removed by the client.

-h|—help — Print a summary of command line options.

-i — Tells **winbindd** to not become a daemon and detach from the current terminal. This option is used by developers when interactive debugging of **winbindd** is required. **winbindd** also logs to standard output, as if the **-S** parameter had been given.

-n — Disable caching. This means winbindd will always have to wait for a response from the domain controller before it can respond to a client and this thus makes things slower. The results will however be more accurate, since results from the cache might not be up-to-date. This might also temporarily hang winbindd if the DC doesn't respond.

-Y — Single daemon mode. This means winbindd will run as a single process (the mode of operation in Samba-2.2). Winbindd's default behavior is to launch a child process that is responsible for updating expired cache entries.

NAME AND ID RESOLUTION

Users and groups on a Windows NT server are assigned a relative id (rid) which is unique for the domain when the user or group is created. To convert the Windows NT user or group into a UNIX user or group, a mapping between rids and UNIX user and group ids is required. This is one of the jobs that **winbindd** performs.

As winbindd users and groups are resolved from a server, user and group ids are allocated from a specified range. This is done on a first come, first served basis, although all existing users and groups will be mapped as soon as a client performs a user or group enumeration

command. The allocated UNIX ids are stored in a database file under the Samba lock directory and will be remembered.

WARNING: The rid to UNIX id database is the only location where the user and group mappings are stored by winbindd. If this file is deleted or corrupted, there is no way for winbindd to determine which user and group ids correspond to Windows NT user and group rids.

CONFIGURATION

Configuration of the **winbindd** daemon is done through configuration parameters in the `smb.conf(5)` file. All parameters should be specified in the `[global]` section of `smb.conf`.

- *winbind separator*
- *idmap uid*
- *idmap gid*
- *winbind cache time*
- *winbind enum users*
- *winbind enum groups*
- *template homedir*
- *template shell*
- *winbind use default domain*

EXAMPLE SETUP

To setup winbindd for user and group lookups plus authentication from a domain controller use something like the following setup. This was tested on a RedHat 6.2 Linux box.

In `/etc/nsswitch.conf` put the following:

```
passwd:    files winbind
group:     files winbind
```

In `/etc/pam.d/*` replace the *auth* lines with something like this:

```
auth required /lib/security/pam_securetty.so
auth required /lib/security/pam_nologin.so
auth sufficient /lib/security/pam_winbind.so
auth required /lib/security/pam_pwdb.so use_first_pass shadow nullok
```

Note in particular the use of the *sufficient* keyword and the *use_first_pass* keyword.

Now replace the account lines with this:

account required /lib/security/pam.winbind.so

The next step is to join the domain. To do that use the **net** program like this:

```
net join -S PDC -U Administrator
```

The username after the **-U** can be any Domain user that has administrator privileges on the machine. Substitute the name or IP of your PDC for "PDC".

Next copy `libnss.winbind.so` to `/lib` and `pam.winbind.so` to `/lib/security`. A symbolic link needs to be made from `/lib/libnss.winbind.so` to `/lib/libnss.winbind.so.2`. If you are using an older version of `glibc` then the target of the link should be `/lib/libnss.winbind.so.1`.

Finally, setup a `smb.conf(5)` containing directives like the following:

```
[global]
  winbind separator = +
  winbind cache time = 10
  template shell = /bin/bash
  template homedir = /home/%D/%U
  idmap uid = 10000-20000
  idmap gid = 10000-20000
  workgroup = DOMAIN
  security = domain
  password server = *
```

Now start `winbindd` and you should find that your user and group database is expanded to include your NT users and groups, and that you can login to your UNIX box as a domain user, using the `DOMAIN+user` syntax for the username. You may wish to use the commands `getent passwd` and `getent group` to confirm the correct operation of `winbindd`.

NOTES

The following notes are useful when configuring and running **winbindd**:

`nmbd(8)` must be running on the local machine for **winbindd** to work. **winbindd** queries the list of trusted domains for the Windows NT server on startup and when a `SIGHUP` is received. Thus, for a running **winbindd** to become aware of new trust relationships between servers, it must be sent a `SIGHUP` signal.

PAM is really easy to misconfigure. Make sure you know what you are doing when modifying PAM configuration files. It is possible to set up PAM such that you can no longer log into your system.

If more than one UNIX machine is running **winbindd**, then in general the user and groups ids allocated by `winbindd` will not be the same. The user and group ids will only be valid for the local machine.

If the the Windows NT RID to UNIX user and group id mapping file is damaged or destroyed then the mappings will be lost.

SIGNALS

The following signals can be used to manipulate the **winbindd** daemon.

SIGHUP — Reload the `smb.conf(5)` file and apply any parameter changes to the running version of `winbindd`. This signal also clears any cached user and group information. The list of other domains trusted by `winbindd` is also reloaded.

SIGUSR1 — The `SIGUSR1` signal will cause **winbindd** to write status information to the `winbind` log file including information about the number of user and group ids allocated by **winbindd**.

Log files are stored in the filename specified by the log file parameter.

FILES

`/etc/nsswitch.conf(5)` — Name service switch configuration file.

`/tmp/.winbindd/pipe` — The UNIX pipe over which clients communicate with the **winbindd** program. For security reasons, the `winbind` client will only attempt to connect to the `winbindd` daemon if both the `/tmp/.winbindd` directory and `/tmp/.winbindd/pipe` file are owned by root.

`$LOCKDIR/winbindd_privilaged/pipe` — The UNIX pipe over which “*privilaged*” clients communicate with the **winbindd** program. For security reasons, access to some `winbindd` functions - like those needed by the **ntlm_auth** utility - is restricted. By default, only users in the ‘root’ group will get this access, however the administrator may change the group permissions on `$LOCKDIR/winbindd_privilaged` to allow programs like ‘squid’ to use `ntlm_auth`. Note that the `winbind` client will only attempt to connect to the `winbindd` daemon if both the `$LOCKDIR/winbindd_privilaged` directory and `$LOCKDIR/winbindd_privilaged/pipe` file are owned by root.

`/lib/libnss_winbind.so.X` — Implementation of name service switch library.

`$LOCKDIR/winbindd_idmap.tdb` — Storage for the Windows NT rid to UNIX user and group id mapping. The lock directory is specified when Samba is initially compiled using the `--with-lockdir` option. This directory is by default `/usr/local/samba/var/locks`.

`$LOCKDIR/winbindd_cache.tdb` — Storage for cached user and group information.

SEE ALSO

`nsswitch.conf`(5), `Samba`(7), `wbinfo`(8), `smb.conf`(5)

THE GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation’s software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
 - (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

- (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components

(compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

one line to give the program’s name and a brief idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) yyyy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type
‘show w’.
This is free software, and you are welcome to redistribute it under certain conditions; type ‘show c’ for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
‘Gnomovision’ (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GLOSSARY

Access Control List (ACL)

A detailed list of permissions granted to users or groups with respect to file and network resource access. See Chapter 12, *File, Directory and Share Access Controls*, for details.

Active Directory Service (ADS)

A service unique to Microsoft Windows 200x servers that provides a centrally managed directory for management of user identities, and computer objects, as well as the permissions each user or computer may be granted to access distributed network resources. ADS uses Kerberos-based authentication and LDAP over Kerberos for directory access.

Common Internet File System (CIFS)

The new name for SMB. Microsoft renamed the SMB protocol to CIFS during the Internet hype in the nineties. At about the time that the SMB protocol was renamed to CIFS, an additional dialect of the SMB protocol was in development. The need for the deployment of the NetBIOS layer was also removed, thus paving the way for use of the SMB protocol natively over TCP/IP (known as NetBIOSless SMB or “*naked*” TCP transport).

Common UNIX Printing System (CUPS)

A recent implementation of a high capability printing system for UNIX developed by ¹. The design objective of CUPS was to provide a rich print processing system that has built-in intelligence that is capable of correctly rendering (processing) a file that is submitted for printing even if it was formatted for an entirely different printer.

Domain Master Browser (DMB)

The Domain Master Browser maintains a list of all the servers that have announced their services within a given workgroup or NT domain. See Section 9.4.1 for details.

Domain Name Service (DNS)

A protocol by which computer host names may be resolved to the matching IP address/es. DNS is implemented by the Berkeley Internet Name Daemon. There exists

¹<http://www.easysw.com/>

a recent version of DNS that allows dynamic name registration by network clients or by a DHCP server. This recent protocol is known as Dynamic DNS (DDNS).

Dynamic Host Configuration Protocol (DHCP)

A protocol that was based on the BOOTP protocol that may be used to dynamically assign an IP address, from a reserved pool of addresses, to a network client or device. Additionally, DHCP may assign all network configuration settings and may be used to register a computer name and its address with a Dynamic DNS server.

Extended Metafile Format (EMF) An intermediate file format used by Microsoft Windows-based servers and clients. EMF files may be rendered into a page description language by a print processor.

Graphical Device Interface (GDI)

Device Independent format for printing used by Microsoft Windows. It is quite similar to what PostScript is for UNIX. Printing jobs are first generated in GDI and then converted to a device-specific format. See Section 18.4.1 for details.

Group Identifier (GID)

The UNIX system Group Identifier; on older systems a 32-bit unsigned integer and on newer systems an unsigned 64-bit integer. The GID is used in UNIX-like operating systems for all group level access control.

Internet Print Protocol (IPP)

An IETF standard for network printing. CUPS implements IPP.

Key Distribution Center (KDC)

The Kerberos authentication protocol makes use of security keys (also called a ticket) by which access to network resources is controlled. The issuing of Kerberos tickets is effected by a KDC.

NetBIOS Extended User Interface (NetBEUI)

Very simple network protocol invented by IBM and Microsoft. It is used to do NetBIOS over ethernet with low overhead. NetBEUI is a non-routable protocol.

Network Basic Input/Output System (NetBIOS)

NetBIOS is a simple application programming interface (API) invented in the eighties that allows programs to send data to certain network names. NetBIOS is always run over another network protocol such as IPX/SPX, TCP/IP, or Logical Link Control (LLC). NetBIOS run over LLC is best known as NetBEUI (The NetBIOS Extended User Interface — a complete misnomer!).

NetBT (NBT)

Protocol for transporting NetBIOS frames over TCP/IP. Uses ports 137, 138 and 139. NetBT is a fully routable protocol.

Local Master Browser (LMB)

The Local Master Browser maintains a list of all servers that have announced themselves within a given workgroup or NT domain on a particular broadcast isolated subnet. See Section 9.4.1 for details.

Printer Command Language (PCL)

A printer page description language that was developed by Hewlett Packard and is in common use today.

Portable Document Format (PDF) A highly compressed document format, based on postscript, used as a document distribution format that is supported by Web browsers as well as many applications. Adobe also distribute an application called “*acrobat*” which is a PDF reader.

Page Description Language (PDL)

A language for describing the layout and contents of a printed page. The best-known PDLs are Adobe PostScript and Hewlett-Packard PCL (Printer Control Language), both of which are used to control laser printers.

PostScript Printer Description (PPD)

PPD's specify and control options supported by postscript printers, such as duplexing, stapling, DPI, ... See also Section 18.4.4. PPD files can be read by printing applications to enable correct postscript page layout for a particular postscript printer.

Server Message Block (SMB)

SMB was the original name of the protocol ‘spoken’ by Samba. It was invented in the eighties by IBM and adopted and extended further by Microsoft. Microsoft renamed the protocol to CIFS during the Internet hype in the nineties.

User Identifier (UID)

The UNIX system User Identifier; on olders systems a 32-bit unsigned integer and on newer systems an unsigned 64-bit integer. The UID is used in UNIX-like operating systems for all user level access control.

Universal Naming Convention (UNC)

A syntax for specifying the location of network resources (such as file shares). The UNC syntax was developed in the early days of MS DOS 3.x and is used internally by the SMB protocol.

SUBJECT INDEX

- valid, 590
- /etc/cups/mime.convs, 258
- /etc/cups/mime.types, 258
- /etc/host.conf, 401
- /etc/hosts, 401
- /etc/krb5.conf, 76
- /etc/nsswitch.conf, 402
- /etc/openldap/slapd.conf, 26
- 8.3 file names, 161

- abort shutdown script, 504
- Account Controls, 363
- ACLs, 159
 - File System, 162
 - POSIX, 159, 160
 - share, 160
 - Windows, 160
- Active Directory, 75
- add group script, 157, 505
- add machine script, 55, 70, 80, 95, 505
- add printer command, 243
- add printer wizard, 259
- add share command, 506
- add user script, 130, 506
- add user to group script, 507
- addprinter command, 505
- admin users, 165, 175, 507
- Administrator, 153
- ADS|see{Active Directory}, 75
- AFS, 419
- algorithmic rid base, 507
- allow hosts, 508
- allow trusted domains, 508
- Amanda, 414
- anonymous
 - print server, 14
 - read-write server, 14
- announce as, 508
- announce version, 508
- application/cups.vnd-postscript, 292
- application/octet-stream, 258, 267, 276
- application/pdf, 266, 267
- application/postscript, 292
- application/vnd.cups-raster, 277
- application/vnd.cups-raw, 258
- auth methods, 145, 429, 508
- auto services, 509
- available, 509

- BackupPC, 413
- bad hardware, 123
- BIND, 492
- bind interfaces only, 509
- block size, 510
- blocking locks, 510
- BOBS, 415
- brlock.tdb|see{TDB}, 308
- browsable, 210, 510
- browse list, 104, 116, 510
- browseable, 210, 216, 218, 223, 510
- browsing problems, 122

- case sensitive, 167, 370, 510
- casesignames, 511
- change notify timeout, 511
- change share command, 511
- chpass, 69
- client lanman auth, 511
- client ntlmv2 auth, 512
- client use spnego, 80, 512
- comment, 216, 217, 223, 512
- config file, 512
- Config.POL, 360
- configure, 471

- connections.tdb|see{TDB}, 308
- copy, 513
- core files, 466
- create mask, 166, 172, 513
- create mode, 513
- csc policy, 167, 513
- CUPS
 - Page Accounting, 317
 - quotas, 317
- CUPS-PPD, 311
- cupssaddmb, 259, 287, 290, 293, 295–297
- cupssomatic, 264, 265, 274, 277, 278, 311
- CVS, 469
 - web, 469

- daemon, 475
- DDK, 286, 289
- deadtime, 514
- debug, 466
- debug hires timestamp, 514
- debug level, 459, 488
- debug pid, 514
- debug timestamp, 514
- debug uid, 515
- debuglevel, 466, 514
- default, 515
- default case, 167, 515
- default devmode, 515
- default profile, 377, 384
- default service, 515
- delete group script, 516
- delete printer command, 243
- delete readonly, 516
- delete roaming profiles, 382
- delete share command, 517
- delete user from group script, 517
- delete user script, 517
- delete veto files, 517
- deleprinter command, 516
- deny hosts, 518
- dfree command, 518
- DFS|see{MS-DFS, Distributed File Systems}, 421
- DHCP, 400
- diff, 467
- directory, 518
- directory mask, 166, 519
- directory mode, 519
- directory security mask, 172, 519
- Directory Separators, 161
- disable netbios, 520
- disable spoolss, 520
- display charset, 410, 446, 520
- Distributed File Systems, 419
- DNS, 105, 106, 406
 - Active Directory, 106
 - Dynamic, 400, 492
- dns proxy, 104, 520
- domain admin group, 151
- Domain Admins group, 153
- domain logons, 47, 521
- domain master, 50, 64, 104, 111, 521
- Domain Member, 34
 - joining, 34
- domain security, 42
- Domain Users group, 157
- dont descend, 167, 521
- dos charset, 410, 522
- dos filemode, 166, 522
- dos filetime resolution, 167, 522
- dos filetimes, 167, 522
- Drive Identification, 161

- editreg, 364
- EMF, 260, 281, 282
- encrypt passwords, 73, 134, 394, 423, 457, 522
- encrypted passwords, 125, 127, 144
- enhanced browsing, 104, 523
- enumports command, 246, 523
- EPM|see{ESP meta packager}, 288
- ESC/P, 282
- ESP
 - Ghostscript, 265, 266, 278
 - meta packager, 288
 - Print Pro, 290
- Event Viewer, 353
- exec, 523
- Extended Attributes, 159

- fake directory create times, 523
- fake oplocks, 167, 524
- File Naming Conventions, 162
- File System, 160
 - case sensitivity, 161
 - feature comparison, 161

- UNIX, 160
- Windows, 160
- flush name cache, 122
- follow symlinks, 524
- foomatic, 264, 265, 274, 278, 310, 311
- foomatic-rip, 277, 310, 312
- force create mode, 166, 172, 524
- force directory mode, 166, 172, 525
- force directory security mode, 166, 172, 525
- force group, 164, 165, 526
- force security mode, 166, 172, 526
- force user, 164, 165, 175, 182, 526
- fstype, 527
- ftp, 470

- gdb, 466
- GDI, 260, 281, 282
- genlogon.pl, 356
- get quota command, 527
- getwd cache, 528
- GhostScript, 262
- Ghostsript
 - ESP|see{ESP GhostScript}, 263
- GhostScript|see{PostScript}, 261
- GID, 151, 153
- GPG, 470
- GPOs, 359, 361–363, 380
- group, 528
- group policies, 359
- group policy objects|see{GPOs}, 359
- group profiles, 376
- groupadd, 152
- groupdel, 152
- groups
 - domain, 154
 - mapping, 151
 - nested, 157
- guest account, 118, 122, 217, 528
- guest ok, 165, 217, 218, 223, 528
- guest only, 529

- hide dot files, 167, 529
- hide files, 167, 529
- hide local users, 529
- hide special files, 529
- hide unreadable, 166, 530
- hide unwriteable files, 166, 530

- homedir map, 530
- host msdfs, 203, 530
- hostname lookups, 530
- hosts allow, 192, 218, 531
- hosts deny, 192, 218, 532
- hosts equiv, 532

- idmap backend, 64, 532
- idmap gid, 151, 351, 397, 532, 665
- idmap uid, 151, 351, 397, 533, 665
- ifconfig, 474
- imprints, 259
- include, 533
- inetd, 454, 473
- inherit acls, 533
- inherit permissions, 533
- initGroups.sh, 25, 156, 437
- Interdomain Trusts, 197
 - Completing, 198
 - creating, 198
 - Facilities, 199
- interfaces, 111, 118, 455, 474, 533
- invalid users, 164, 165, 534
- IPP, 295
- ISC
 - DHCP, 491
 - DNS, 491

- KDC, 75
- keepalive, 535
- Kerberos, 75
 - /etc/krb5.conf, 76
- kernel oplocks, 535
- kinit, 76

- lanman auth, 535
- large readwrite, 536
- ldap admin dn, 63, 79, 139, 536, 654
- ldap delete dn, 139, 536
- ldap filter, 139, 536
- ldap group suffix, 139, 536
- ldap idmap suffix, 63, 79, 139, 536
- ldap machine suffix, 139, 536
- ldap passwd sync, 139, 143, 536
- ldap server, 537
- ldap ssl, 139, 141, 537
- ldap suffix, 63, 139, 537
- ldap user suffix, 139, 537
- level2 oplocks, 537

- libnss_wins.so, 402
- Links
 - hard, 162
 - soft, 162
- Linuxprinting.org, 309
- lm announce, 104, 538
- lm interval, 104, 538
- LMB|see{Local Master Browser}, 105, 112
- LMHOSTS, 404
- load printers, 211, 212, 215, 539
- local master, 104, 108, 539, 636
- Local Master Browser, 105, 112
- lock dir, 539
- lock directory, 539
- lock spin count, 539
- lock spin time, 540
- locking, 178, 539
- locking.tdb|see{TDB}, 308
- log file, 540
- log files
 - monitoring, 452
- log level, 80, 118, 332, 466, 540, 599, 601, 610, 615, 625, 635, 641, 643, 646, 656, 657, 664
- logon drive, 372, 540
- logon home, 142, 368, 369, 372, 374, 540
- logon path, 147, 369–372, 374, 541
- logon script, 147, 541
- lpadmin, 309, 317
- lppause command, 255, 283, 322, 542
- lpq cache time, 216, 543
- lpq command, 255, 322, 543
- lpresume command, 255, 322, 543
- lprm command, 255, 322, 544
- lpstat, 307
- Lustre, 419

- MAC Addresses, 400
- machine password timeout, 544
- Machine Trust Accounts, 44, 64, 68
 - creating, 68
- magic output, 544
- magic script, 545
- make, 471
- mandatory profiles, 376
- mangle case, 545
- mangle prefix, 547
- mangled map, 545
- mangled names, 545
- mangled stack, 546
- mangling char, 547
- mangling method, 547
- map acl inherit, 547
- map archive, 547
- map hidden, 548
- map system, 548
- map to guest, 224, 243, 325, 548
- max connections, 549
- max disk size, 549
- max log size, 549, 646
- max mux, 549
- max open files, 549
- max print jobs, 550
- max protocol, 550
- max reported print jobs, 550
- max smbd processes, 550
- max ttl, 551
- max wins ttl, 551
- max xmit, 488, 551
- message command, 551
- messages.tdb|see{TDB}, 308
- MIME, 266–268, 276
 - filters, 266
 - raw, 16, 84, 258
- min passwd length, 552
- min password length, 552
- min print space, 552
- min protocol, 552
- min wins ttl, 553
- minimal configuration, 5
- MS-DFS, 421
- msdfs proxy, 553
- msdfs root, 203, 553

- name cache timeout, 553
- name resolve order, 104, 553, 600, 634
- nbtstat, 404
- net
 - groupmap, 25, 154, 437
 - rpc, 20, 34, 58, 436
- NetBIOS, 103, 105, 399, 403
- netbios aliases, 554
- netbios name, 554, 598, 602, 616, 624
- netbios scope, 555
- NetBIOS-less, 105
- Nexus.exe, 43, 70, 353

- NFS, 419
- nis homedir, 555
- nmblookup, 404
- NoMachine.Com, 354
- non unix account range, 555
- nt acl support, 166, 169–171, 484, 485, 555
- nt pipe support, 556
- nt status support, 556
- NTConfig.POL, 360, 363, 364, 378
- ntdrivers.tdb|see{TDB}, 308
- ntforms.tdb|see{TDB}, 308
- NTFS, 160
- ntlm auth, 556
- ntprinters.tdb|see{TDB}, 308
- NTUser.DAT, 364
- null passwords, 556

- obey pam restrictions, 393, 394, 556, 647
- office server, 17
- only guest, 556
- only user, 165, 195, 556
- OpenGFS, 419
- oplock break wait time, 182, 185, 557
- oplock contention limit, 182, 557
- oplocks, 557
- os level, 104, 108–111, 558
- os2 driver map, 482, 558

- page_log, 318
- pam password change, 558
- panic action, 558
- paranoid server security, 558
- passdb backend, 17, 68, 127, 133, 139, 143, 145, 146, 394, 423, 429, 559
- passwd chat, 559
- passwd chat debug, 560
- passwd program, 560
- password level, 38, 456, 483, 489, 561
- password server, 36, 53, 73, 75, 457, 561
- patch, 467
- path, 216–218, 222–224, 283, 321, 328, 456, 563
- PCL, 260, 282, 284
- pdbedit, 24, 126, 131–133, 145, 429, 436, 439
- PDF, 260, 264
- pdf, 267
- PDL, 260, 262

- permissions
 - file/directory ACLs, 169
 - share, 164
 - share ACLs, 165
 - UNIX file and directory, 160
- PGP, 471
- pid directory, 563
- PJL, 284, 292, 318
- point 'n' print, 257, 259, 274, 293, 297, 306
- posix locking, 563
- postexec, 564
- PostScript, 260, 261, 268, 282, 284, 286, 287
 - RIP, 261
- PostScript|see{Ghostscript}, 259
- PPD, 262, 263, 277, 285, 286, 298
 - CUPS|see{CUPS-PPD}, 311
- preexec, 564
- preexec close, 564
- prefered master, 564
- preferred master, 104, 108, 110, 111, 457, 564
- preload, 565
- preload modules, 565
- preserve case, 370, 565
- print command, 215, 218–220, 255, 283, 322, 566
- print ok, 568
- printable, 216–218, 565
- printcap, 219, 254, 256, 321, 565
- printcap name, 215, 566
- printer, 567
- printer admin, 215–217, 224, 226, 235, 236, 238, 240, 242, 256, 301, 327, 568
- printer name, 568
- Printers folder, 292, 298, 306
- printing, 215, 218–220, 254–256, 283, 321, 322, 568
- printing.tdb|see{TDB}, 308
- private dir, 568
- profile acls, 568
- protocol, 569
- public, 217, 569

- queue resume command, 255
- queuepause command, 255, 569
- queueresume command, 569

- raw printing, 16, 84, 257, 258
- read bmpx, 570
- read list, 165, 570
- read only, 167, 217, 224, 570
 - server, 12
- read raw, 489, 570
- read size, 488, 571
- realm, 571
- Relative Identifier|see{RID}, 154
- remote announce, 105, 107, 112, 117, 571
- remote browse sync, 105, 107, 112, 572
- replication, 43
 - browse lists, 118
 - SAM, 45, 57–59, 63, 65
 - WINS, 105, 114
- restrict anonymous, 572
- RID, 154
- roaming profiles, 369
- root, 572
- root dir, 573
- root directory, 573
- root postexec, 573
- root preexec, 436, 573
- root preexec close, 573
- rpcclient
 - adddriver, 294, 295, 299–301, 304
 - enumdrivers, 299, 305
 - enumports, 299
 - enumprinters, 299, 302, 305–307
 - getdriver, 300, 303, 305
 - getprinter, 300, 303, 305, 307
 - setdriver, 292, 294, 295, 299, 302, 305
- rsync, 470
- rundll32, 357
- SAM, 43, 44, 67, 339
- SAM backend
 - LDAP, 57
 - ldapsam, 58, 126, 130, 134
 - ldapsam_compat, 125
 - mysqlsam, 126, 143
 - non-LDAP, 57
 - smbpasswd, 125, 134
 - tidsam, 58, 126, 134
 - xmlsam, 126, 145
- samba-ldap-init.ldif, 27
- schannel, 55
- SCSCI, 421
- secrets.tdb|see{TDB}, 308
- security, 32, 34, 36, 39, 46, 53, 73–75, 293, 324, 429, 457, 483, 573
- security mask, 166, 172, 577
- Security Mode, 32
- Server Manager, 69, 70, 353
- server schannel, 577
- server signing, 577
- server string, 578
- Server Type, 32
 - Domain Controller, 23
 - Domain Member, 19, 34, 64, 67
 - Stand-alone, 12
- sessionid.tdb|see{TDB}, 308
- set directory, 578
- set primary group script, 578
- set quota command, 578
- share modes, 579
- share.info.tdb|see{TDB}, 308
- short preserve case, 167, 370, 579
- Short-Cuts, 162
- show add printer wizard, 215, 242, 580
- shutdown script, 580
- SID, 42, 54, 62, 127, 153, 375, 436
- signing, 55
- simple configuration, 6
- Single Sign On, 291
- slow browsing, 122
- smb passwd file, 581
- smb ports, 581
- smbclient, 78, 455
- smbgrpadd.sh, 155
- socket address, 581
- socket options, 487, 488, 581
- source environment, 582
- spooling
 - central, 257
 - peer-to-peer, 257
- spooling-only, 257
- SRVTOOLS.EXE, 70, 353
- stat cache, 583
- stat cache size, 583
- strict allocate, 583
- strict locking, 178, 583
- strict sync, 583
- strip dot, 584
- swat, 7
 - enable, 444

- security, 445
- sync always, 584
- syslog, 584
- syslog only, 584
- System Policy Editor, 360, 362
- TDB, 308
 - backing up|see{tdbbackup}, 309
- tdbbackup, 309
- template homedir, 348, 584, 665
- template shell, 584, 665
- testparm, 452
- text/plain, 267
- time offset, 585
- time server, 585
- timestamp logs, 585
- total print jobs, 215, 585
- UDP, 105
- UID, 151
- unexpected.tdb|see{TDB}, 308
- unicode, 585
- unix charset, 410, 585
- unix extensions, 585
- unix password sync, 586
- update encrypted, 586
- use client driver, 216, 586
- use mmap, 587
- use sendfile, 589
- use spnego, 590
- user, 34, 456, 587
- User Accounts
 - Adding/Deleting, 131, 132, 139
- User Management, 131, 132, 139
- User Manager, 200, 201, 353
- useradd, 69
- username, 165, 587
- username level, 38, 588
- username map, 71, 588
- users, 195, 589
- utmp, 590
- utmp directory, 590
- valid users, 164, 165, 456, 590
- veto files, 167, 591
- veto oplock files, 591
- vfs object, 592
- vfs objects, 331, 592
- vipw, 69
- volume, 592
- WebClient, 123
- wide links, 592
- winbind cache time, 592, 665
- winbind enum groups, 592, 665
- winbind enum users, 592, 665
- winbind gid, 593
- winbind separator, 344, 593, 665
- winbind uid, 593
- winbind use default domain, 593, 665
- winbindd, 64
- windows registry settings
 - default profile locations, 379, 381
 - profile path, 371
 - roaming profiles, 369
- WINS, 104, 105, 406, 492
- wins hook, 104, 593
- wins partners, 594
- wins proxy, 104, 594
- wins server, 104, 113, 114, 594
- wins support, 104, 113, 114, 595, 636
- workgroup, 53, 73, 117, 595
- writable, 595
- write cache size, 595
- write list, 165, 224, 596
- write ok, 596
- write raw, 489, 596
- writable, 217, 218, 595
- wtmp directory, 596
- WYSIWYG, 260
- X Window System, 260
- xinetd, 474
- xinetd|see{inetd}, 454
- Xprint, 260