

SAMBA Project Documentation

Table of Contents

<u>SAMBA Project Documentation</u>	1
<u>SAMBA Team</u>	1
<u>Abstract</u>	2
<u>How to Install and Test SAMBA</u>	7
<u>Step 0: Read the man pages</u>	7
<u>Step 1: Building the Binaries</u>	7
<u>Step 2: The all important step</u>	7
<u>Step 3: Create the smb configuration file</u>	8
<u>Step 4: Test your config file with testparm</u>	8
<u>Step 5: Starting the smbd and nmbd</u>	8
<u>Step 5a: Starting from inetd.conf</u>	8
<u>Step 5b. Alternative: starting it as a daemon</u>	9
<u>Step 6: Try listing the shares available on your server</u>	10
<u>Step 7: Try connecting with the unix client</u>	10
<u>Step 8: Try connecting from a DOS, WfWg, Win9x, WinNT, Win2k, OS/2, etc... client</u>	10
<u>What If Things Don't Work?</u>	10
<u>Diagnosing Problems</u>	11
<u>Scope IDs</u>	11
<u>Choosing the Protocol Level</u>	11
<u>Printing from UNIX to a Client PC</u>	11
<u>Locking</u>	12
<u>Mapping Usernames</u>	12
<u>Diagnosing your samba server</u>	13
<u>Introduction</u>	13
<u>Assumptions</u>	13
<u>Tests</u>	13
<u>Test 1</u>	13
<u>Test 2</u>	14
<u>Test 3</u>	14
<u>Test 4</u>	15
<u>Test 5</u>	15
<u>Test 6</u>	15
<u>Test 7</u>	16
<u>Test 8</u>	16
<u>Test 9</u>	17
<u>Test 10</u>	17
<u>Test 11</u>	17
<u>Still having troubles?</u>	17
<u>Integrating MS Windows networks with Samba</u>	18
<u>Agenda</u>	18
<u>Name Resolution in a pure Unix/Linux world</u>	18
<u>/etc/hosts</u>	18
<u>/etc/resolv.conf</u>	19
<u>/etc/host.conf</u>	19

Table of Contents

<u>Integrating MS Windows networks with Samba</u>	
<u>/etc/nsswitch.conf</u>	19
<u>Name resolution as used within MS Windows networking</u>	20
<u>The NetBIOS Name Cache</u>	21
<u>The LMHOSTS file</u>	22
<u>HOSTS file</u>	23
<u>DNS Lookup</u>	23
<u>WINS Lookup</u>	23
<u>How browsing functions and how to deploy stable and dependable browsing using Samba</u>	24
<u>MS Windows security options and how to configure Samba for seamless integration</u>	25
<u>Use MS Windows NT as an authentication server</u>	26
<u>Make Samba a member of an MS Windows NT security domain</u>	26
<u>Configure Samba as an authentication server</u>	27
<u>Conclusions</u>	27
<u>Configuring PAM for distributed but centrally managed authentication</u>	29
<u>Samba and PAM</u>	29
<u>Distributed Authentication</u>	30
<u>PAM Configuration in smb.conf</u>	31
<u>Hosting a Microsoft Distributed File System tree on Samba</u>	32
<u>Instructions</u>	32
<u>Notes</u>	33
<u>UNIX Permission Bits and Windows NT Access Control Lists</u>	34
<u>Viewing and changing UNIX permissions using the NT security dialogs</u>	34
<u>How to view file security on a Samba share</u>	34
<u>Viewing file ownership</u>	34
<u>Viewing file or directory permissions</u>	35
<u>File Permissions</u>	35
<u>Directory Permissions</u>	35
<u>Modifying file or directory permissions</u>	36
<u>Interaction with the standard Samba create mask parameters</u>	36
<u>Interaction with the standard Samba file attribute mapping</u>	38
<u>Printing Support in Samba 2.2.x</u>	39
<u>Introduction</u>	39
<u>Configuration</u>	39
<u>Creating [print\$]</u>	40
<u>Setting Drivers for Existing Printers</u>	41
<u>Support a large number of printers</u>	42
<u>Adding New Printers via the Windows NT APW</u>	42
<u>Samba and Printer Ports</u>	43
<u>The Imprints Toolset</u>	44
<u>What is Imprints?</u>	44
<u>Creating Printer Driver Packages</u>	44
<u>The Imprints server</u>	44
<u>The Installation Client</u>	44

Table of Contents

<u>Printing Support in Samba 2.2.x</u>	
<u>Migration to from Samba 2.0.x to 2.2.x</u>	45
<u>Debugging Printing Problems</u>	47
<u>Introduction</u>	47
<u>Debugging printer problems</u>	48
<u>What printers do I have?</u>	48
<u>Setting up printcap and print servers</u>	49
<u>Job sent, no output</u>	49
<u>Job sent, strange output</u>	50
<u>Raw PostScript printed</u>	50
<u>Advanced Printing</u>	50
<u>Real debugging</u>	51
<u>Security levels</u>	52
<u>Introduction</u>	52
<u>More complete description of security levels</u>	52
<u>security = domain in Samba 2.x</u>	54
<u>Joining an NT Domain with Samba 2.2</u>	54
<u>Samba and Windows 2000 Domains</u>	55
<u>Why is this better than security = server?</u>	55
<u>Unified Logons between Windows NT and UNIX using Winbind</u>	57
<u>Abstract</u>	57
<u>Introduction</u>	57
<u>What Winbind Provides</u>	57
<u>Target Uses</u>	58
<u>How Winbind Works</u>	58
<u>Microsoft Remote Procedure Calls</u>	58
<u>Name Service Switch</u>	59
<u>Pluggable Authentication Modules</u>	59
<u>User and Group ID Allocation</u>	60
<u>Result Caching</u>	60
<u>Installation and Configuration</u>	60
<u>Introduction</u>	60
<u>Requirements</u>	61
<u>Testing Things Out</u>	61
<u>Limitations</u>	68
<u>Conclusion</u>	69
<u>How to Configure Samba 2.2 as a Primary Domain Controller</u>	70
<u>Prerequisite Reading</u>	70
<u>Background</u>	70
<u>Configuring the Samba Domain Controller</u>	71
<u>Creating Machine Trust Accounts and Joining Clients to the Domain</u>	72
<u>Manual Creation of Machine Trust Accounts</u>	72
<u>"On-the-Fly" Creation of Machine Trust Accounts</u>	73

Table of Contents

<u>How to Configure Samba 2.2 as a Primary Domain Controller</u>	
<u>Joining the Client to the Domain</u>	73
<u>Common Problems and Errors</u>	74
<u>System Policies and Profiles</u>	75
<u>What other help can I get?</u>	76
<u>Domain Control for Windows 9x/ME</u>	79
<u>Configuration Instructions: Network Logons</u>	80
<u>Configuration Instructions: Setting up Roaming User Profiles</u>	80
<u>DOMAIN CONTROL.txt : Windows NT Domain Control & Samba</u>	84
<u>How to Act as a Backup Domain Controller in a Purely Samba Controlled Domain</u>	87
<u>Prerequisite Reading</u>	87
<u>Background</u>	87
<u>What qualifies a Domain Controller on the network?</u>	87
<u>How does a Workstation find its domain controller?</u>	88
<u>When is the PDC needed?</u>	88
<u>Can Samba be a Backup Domain Controller?</u>	88
<u>How do I set up a Samba BDC?</u>	88
<u>How do I replicate the smbpasswd file?</u>	89
<u>Storing Samba's User/Machine Account information in an LDAP Directory</u>	90
<u>Purpose</u>	90
<u>Introduction</u>	90
<u>Supported LDAP Servers</u>	91
<u>Schema and Relationship to the RFC 2307 posixAccount</u>	91
<u>Configuring Samba with LDAP</u>	92
<u>OpenLDAP configuration</u>	92
<u>Configuring Samba</u>	92
<u>Accounts and Groups management</u>	93
<u>Security and sambaAccount</u>	94
<u>LDAP specials attributes for sambaAccounts</u>	94
<u>Example LDIF Entries for a sambaAccount</u>	95
<u>Comments</u>	96
<u>Improved browsing in samba</u>	97
<u>Overview of browsing</u>	97
<u>Browsing support in samba</u>	97
<u>Problem resolution</u>	97
<u>Browsing across subnets</u>	98
<u>How does cross subnet browsing work ?</u>	98
<u>Setting up a WINS server</u>	101
<u>Setting up Browsing in a WORKGROUP</u>	102
<u>Setting up Browsing in a DOMAIN</u>	103
<u>Forcing samba to be the master</u>	103
<u>Making samba the domain master</u>	104
<u>Note about broadcast addresses</u>	105
<u>Multiple interfaces</u>	105

Table of Contents

<u>Samba performance issues</u>	106
<u>Comparisons</u>	106
<u>Oplocks</u>	106
<u>Overview</u>	106
<u>Level2 Oplocks</u>	106
<u>Old 'fake oplocks' option – deprecated</u>	107
<u>Socket options</u>	107
<u>Read size</u>	107
<u>Max xmit</u>	108
<u>Locking</u>	108
<u>Share modes</u>	108
<u>Log level</u>	108
<u>Wide lines</u>	109
<u>Read raw</u>	109
<u>Write raw</u>	109
<u>Read prediction</u>	109
<u>Memory mapping</u>	109
<u>Slow Clients</u>	110
<u>Slow Logins</u>	110
<u>Client tuning</u>	110
<u>My Results</u>	111
<u>Samba and other CIFS clients</u>	112
<u>Macintosh clients?</u>	112
<u>OS2 Client</u>	112
<u>How can I configure OS/2 Warp Connect or OS/2 Warp 4 as a client for Samba?</u>	112
<u>How can I configure OS/2 Warp 3 (not Connect), OS/2 1.2, 1.3 or 2.x for Samba?</u>	113
<u>Are there any other issues when OS/2 (any version) is used as a client?</u>	113
<u>How do I get printer driver download working for OS/2 clients?</u>	113
<u>Windows for Workgroups</u>	113
<u>Use latest TCP/IP stack from Microsoft</u>	113
<u>Delete .pwl files after password change</u>	114
<u>Configure WfW password handling</u>	114
<u>Case handling of passwords</u>	114
<u>Windows '95/'98</u>	114
<u>Windows 2000 Service Pack 2</u>	115
<u>HOWTO Access Samba source code via CVS</u>	116
<u>Introduction</u>	116
<u>CVS Access to samba.org</u>	116
<u>Access via CVSweb</u>	116
<u>Access via cvs</u>	116
<u>Reporting Bugs</u>	118
<u>Introduction</u>	118
<u>General info</u>	118
<u>Debug levels</u>	118
<u>Internal errors</u>	119

Table of Contents

Reporting Bugs

<u>Attaching to a running process</u>	119
<u>Patches</u>	119

Group mapping HOWTO.....120

Portability.....121

<u>HPUX</u>	121
<u>SCO Unix</u>	121
<u>DNIX</u>	121

SAMBA Project Documentation

SAMBA Team

Abstract

Last Update : Thu Aug 15 12:48:45 CDT 2002

This book is a collection of HOWTOs added to Samba documentation over the years. I try to ensure that all are current, but sometimes the is a larger job than one person can maintain. The most recent version of this document can be found at <http://www.samba.org/> on the "Documentation" page. Please send updates to jerry@samba.org.

This documentation is distributed under the GNU General Public License (GPL) version 2. A copy of the license is included with the Samba source distribution. A copy can be found on-line at <http://www.fsf.org/licenses/gpl.txt>

Cheers, jerry

Table of Contents

How to Install and Test SAMBA

Step 0: Read the man pages

Step 1: Building the Binaries

Step 2: The all important step

Step 3: Create the smb configuration file.

*Step 4: Test your config file with **testparm***

Step 5: Starting the smbd and nmbd

Step 5a: Starting from inetd.conf

Step 5b. Alternative: starting it as a daemon

Step 6: Try listing the shares available on your server

Step 7: Try connecting with the unix client

Step 8: Try connecting from a DOS, WfWg, Win9x, WinNT, Win2k, OS/2, etc... client

What If Things Don't Work?

Diagnosing Problems

Scope IDs

Choosing the Protocol Level

Printing from UNIX to a Client PC

Locking

Mapping Usernames

Diagnosing your samba server

Introduction

Assumptions

Tests

Test 1

Test 2

Test 3

Test 4

Test 5

Test 6

Test 7

Test 8

Test 9

Test 10

Test 11

[Still having troubles?](#)

[Integrating MS Windows networks with Samba](#)

[Agenda](#)

[Name Resolution in a pure Unix/Linux world](#)

[/etc/hosts](#)

[/etc/resolv.conf](#)

[/etc/host.conf](#)

[/etc/nsswitch.conf](#)

[Name resolution as used within MS Windows networking](#)

[The NetBIOS Name Cache](#)

[The LMHOSTS file](#)

[HOSTS file](#)

[DNS Lookup](#)

[WINS Lookup](#)

[How browsing functions and how to deploy stable and dependable browsing using Samba](#)

[MS Windows security options and how to configure Samba for seamless integration](#)

[Use MS Windows NT as an authentication server](#)

[Make Samba a member of an MS Windows NT security domain](#)

[Configure Samba as an authentication server](#)

[Conclusions](#)

[Configuring PAM for distributed but centrally managed authentication](#)

[Samba and PAM](#)

[Distributed Authentication](#)

[PAM Configuration in smb.conf](#)

[Hosting a Microsoft Distributed File System tree on Samba](#)

[Instructions](#)

[Notes](#)

[UNIX Permission Bits and Windows NT Access Control Lists](#)

[Viewing and changing UNIX permissions using the NT security dialogs](#)

[How to view file security on a Samba share](#)

[Viewing file ownership](#)

[Viewing file or directory permissions](#)

[File Permissions](#)

[Directory Permissions](#)

[Modifying file or directory permissions](#)

[Interaction with the standard Samba create mask parameters](#)

[Interaction with the standard Samba file attribute mapping](#)

[Printing Support in Samba 2.2.x](#)

[Introduction](#)

[Configuration](#)

[Creating \[print\\$\]](#)

[Setting Drivers for Existing Printers](#)

[Support a large number of printers](#)

[Adding New Printers via the Windows NT APW](#)

[Samba and Printer Ports](#)

[The Imprints Toolset](#)

[What is Imprints?](#)

[Creating Printer Driver Packages](#)

[The Imprints server](#)

[The Installation Client](#)

[Migration to from Samba 2.0.x to 2.2.x](#)

Debugging Printing Problems

Introduction

Debugging printer problems

What printers do I have?

Setting up printcap and print servers

Job sent, no output

Job sent, strange output

Raw PostScript printed

Advanced Printing

Real debugging

Security levels

Introduction

More complete description of security levels

security = domain in Samba 2.x

Joining an NT Domain with Samba 2.2

Samba and Windows 2000 Domains

Why is this better than security = server?

Unified Logons between Windows NT and UNIX using Winbind

Abstract

Introduction

What Winbind Provides

Target Uses

How Winbind Works

Microsoft Remote Procedure Calls

Name Service Switch

Pluggable Authentication Modules

User and Group ID Allocation

Result Caching

Installation and Configuration

Introduction

Requirements

Testing Things Out

Limitations

Conclusion

How to Configure Samba 2.2 as a Primary Domain Controller

Prerequisite Reading

Background

Configuring the Samba Domain Controller

Creating Machine Trust Accounts and Joining Clients to the Domain

Manual Creation of Machine Trust Accounts

"On-the-Fly" Creation of Machine Trust Accounts

Joining the Client to the Domain

Common Problems and Errors

System Policies and Profiles

What other help can I get?

Domain Control for Windows 9x/ME

Configuration Instructions: Network Logons

Configuration Instructions: Setting up Roaming User Profiles

DOMAIN CONTROL.txt : Windows NT Domain Control & Samba

How to Act as a Backup Domain Controller in a Purely Samba Controlled Domain

Prerequisite Reading

SAMBA Project Documentation

Background

What qualifies a Domain Controller on the network?

How does a Workstation find its domain controller?

When is the PDC needed?

Can Samba be a Backup Domain Controller?

How do I set up a Samba BDC?

How do I replicate the smbpasswd file?

Storing Samba's User/Machine Account information in an LDAP Directory

Purpose

Introduction

Supported LDAP Servers

Schema and Relationship to the RFC 2307 posixAccount

Configuring Samba with LDAP

OpenLDAP configuration

Configuring Samba

Accounts and Groups management

Security and sambaAccount

LDAP specials attributes for sambaAccounts

Example LDIF Entries for a sambaAccount

Comments

Improved browsing in samba

Overview of browsing

Browsing support in samba

Problem resolution

Browsing across subnets

How does cross subnet browsing work ?

Setting up a WINS server

Setting up Browsing in a WORKGROUP

Setting up Browsing in a DOMAIN

Forcing samba to be the master

Making samba the domain master

Note about broadcast addresses

Multiple interfaces

Samba performance issues

Comparisons

Oplocks

Overview

Level2 Oplocks

Old 'fake oplocks' option – deprecated

Socket options

Read size

Max xmit

Locking

Share modes

Log level

Wide lines

Read raw

Write raw

Read prediction

Memory mapping

Slow Clients

SAMBA Project Documentation

[Slow Logins](#)

[Client tuning](#)

[My Results](#)

[Samba and other CIFS clients](#)

[Macintosh clients?](#)

[OS2 Client](#)

[How can I configure OS/2 Warp Connect or OS/2 Warp 4 as a client for Samba?](#)

[How can I configure OS/2 Warp 3 \(not Connect\), OS/2 1.2, 1.3 or 2.x for Samba?](#)

[Are there any other issues when OS/2 \(any version\) is used as a client?](#)

[How do I get printer driver download working for OS/2 clients?](#)

[Windows for Workgroups](#)

[Use latest TCP/IP stack from Microsoft](#)

[Delete .pwl files after password change](#)

[Configure WfW password handling](#)

[Case handling of passwords](#)

[Windows '95/'98](#)

[Windows 2000 Service Pack 2](#)

[HOWTO Access Samba source code via CVS](#)

[Introduction](#)

[CVS Access to samba.org](#)

[Access via CVSweb](#)

[Access via cvs](#)

[Reporting Bugs](#)

[Introduction](#)

[General info](#)

[Debug levels](#)

[Internal errors](#)

[Attaching to a running process](#)

[Patches](#)

[Group mapping HOWTO](#)

[Portability](#)

[HPUX](#)

[SCO Unix](#)

[DNIX](#)

How to Install and Test SAMBA

Step 0: Read the man pages

The man pages distributed with SAMBA contain lots of useful info that will help to get you started. If you don't know how to read man pages then try something like:

```
$ nroff -man smbd.8 | more
```

Other sources of information are pointed to by the Samba web site, <http://www.samba.org>

Step 1: Building the Binaries

To do this, 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. Then executing

```
root# make
```

will create the binaries. Once it's 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 for a previous version of Samba you might like to know that the old versions of the binaries will be renamed with a ".old" extension. You can go back to the previous version with

```
root# make revert
```

if you find this version a disaster!

Step 2: The all important step

At this stage you must fetch yourself a coffee or other drink you find stimulating. Getting the rest of the install right can sometimes be tricky, so you will probably need it.

If you have installed samba before then you can skip this step.

Step 3: Create the smb configuration file.

There are sample configuration files in the examples subdirectory in the distribution. I suggest you read them carefully so you can see how the options go together in practice. See the man page for all the options.

The simplest useful configuration file would be something like this:

```
[global]
  workgroup = MYGROUP

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

which would allow connections by anyone with an account on the server, using either their login name or "homes" as the service name. (Note that I also set the workgroup that Samba is part of. See BROWSING.txt for details)

Note that **make install** will not install a `smb.conf` file. You need to create it yourself.

Make sure you put the `smb.conf` file in the same place you specified in the `Makefile` (the default is to look for it in `/usr/local/samba/lib/`).

For more information about security settings for the [homes] share please refer to the document `UNIX_SECURITY.txt`.

Step 4: Test your config file with testparm

It's important that you test the validity of your `smb.conf` file using the `testparm` program. If `testparm` runs OK then it will list the loaded services. If not it will give an error message.

Make sure it runs OK and that the services look reasonable before proceeding.

Step 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 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 be in order 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.

Step 5a: Starting from inetd.conf

NOTE; The following will be different if you use NIS or NIS+ to distributed services maps.

SAMBA Project Documentation

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 something 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.

NOTE: Some unixes already have entries like `netbios_ns` (note the underscore) in `/etc/services`. You must either 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 don't know what the broadcast is for your net. **nmbd** tries to determine it at run time, but fails on some unixes. See the section on "testing nmbd" for a method of finding if you need to do this.

!!!WARNING!!! Many unixes only accept around 5 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. If you have installed an earlier version of **nmbd** then you may need to kill `nmbd` as well.

Step 5b. Alternative: starting it as a daemon

To start the server as a daemon you should create a script something like this one, perhaps calling it `startsm`.

```
#!/bin/sh
/usr/local/samba/bin/smbd -D
/usr/local/samba/bin/nmbd -D
```

then make it executable with **chmod +x startsm**

You can then run **startsm** 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 then you may like to look at the `examples/svr4-startup` script to make Samba fit into that system.

Step 6: Try listing the shares available on your server

```
$ smbclient -L yourhostname
```

You should get back a list of shares available on your server. If you don't then something is incorrectly setup. Note that this method can also be used to see what shares are available on other LanManager clients (such as WfWg).

If you choose user level security then 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 `-U%` to the command line. This will not work with non-Samba servers)

Step 7: Try connecting with the unix client

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

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

For example if your unix host is `bambi` and your login name is `fred` you would type:

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

Step 8: Try connecting from a DOS, WfWg, Win9x, WinNT, Win2k, OS/2, etc... client

Try mounting disks. eg:

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

Try printing. eg:

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

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

Celebrate, or send me a bug report!

What If Things Don't Work?

If nothing works and you start to think "who wrote this pile of trash" then I suggest you do step 2 again (and again) till you calm down.

Then you might read the file `DIAGNOSIS.txt` and the `FAQ`. If you are still stuck then try the mailing list or newsgroup (look in the `README` for details). Samba has been successfully installed at thousands of sites

worldwide, so maybe someone else has hit your problem and has overcome it. You could also use the WWW site to scan back issues of the samba-digest.

When you fix the problem PLEASE send me some updates to the documentation (or source code) so that the next person will find it easier.

Diagnosing Problems

If you have installation problems then go to `DIAGNOSIS.txt` to try to find the problem.

Scope IDs

By default Samba uses a blank scope ID. This means all your windows boxes must also have a blank scope ID. If you really want to use a non-blank scope ID then you will need to use the 'netbios scope' `smb.conf` option. All your PCs will need to have the same setting for this to work. I do not recommend scope IDs.

Choosing the Protocol Level

The SMB protocol has many dialects. Currently Samba supports 5, called CORE, COREPLUS, LANMAN1, LANMAN2 and NT1.

You can choose what maximum protocol to support in the `smb.conf` file. The default is NT1 and that is the best for the vast majority of sites.

In older versions of Samba you may have found it necessary to use COREPLUS. The limitations that led to this have mostly been fixed. It is now less likely that you will want to use less than LANMAN1. The only remaining advantage of COREPLUS is that for some obscure reason WfWg preserves the case of passwords in this protocol, whereas under LANMAN1, LANMAN2 or NT1 it uppercases all passwords before sending them, forcing you to use the "password level=" option in some cases.

The main advantage of LANMAN2 and NT1 is support for long filenames with some clients (eg: smbclient, Windows NT or Win95).

See the `smb.conf(5)` manual page for more details.

Note: To support print queue reporting you may find that you have to use TCP/IP as the default protocol under WfWg. For some reason if you leave Netbeui as the default it may break the print queue reporting on some systems. It is presumably a WfWg bug.

Printing from UNIX to a Client PC

To use a printer that is available via a smb-based server from a unix host with LPR you will need to compile the smbclient program. You then need to install the script "smbprint". Read the instruction in smbprint for more details.

There is also a SYSV style script that does much the same thing called `smbprint.sysv`. It contains instructions.

See the CUPS manual for information about setting up printing from a unix host with CUPS to a smb-based server.

Locking

One area which sometimes causes trouble is locking.

There are two types of locking which need to be performed by a SMB server. The first is "record locking" which 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 is 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 can not be fully correct due to 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 $0-2^{31}$, Samba hands this request down to the Unix system. All other locks can not be seen by unix anyway.

Strictly a 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`. It is also almost always unnecessary as clients are supposed to independently make locking calls before reads and writes anyway 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" then it will make lock checking calls on every read and write.

You can also disable by range locking completely using "locking = no". This is useful for those shares that don't support locking or don't need it (such as cdroms). In this case Samba fakes the return codes of locking calls to tell clients that everything is OK.

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`.

Mapping Usernames

If you have different usernames on the PCs and the unix server then take a look at the "username map" option. See the `smb.conf` man page for details.

Diagnosing your samba server

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. I have tried to carefully choose them so later tests only use capabilities verified in the earlier tests.

If you send me an email saying "it doesn't work" and you have not followed this test procedure then you should not be surprised if I ignore your email.

Assumptions

In all of the tests I assume you have a Samba server called BIGSERVER and a PC called ACLIENT both in workgroup TESTGROUP. I also assume the PC is running windows for workgroups with a recent copy of the microsoft tcp/ip stack. Alternatively, your PC may be running Windows 95 or Windows NT (Workstation or Server).

The procedure is similar for other types of clients.

I also assume 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 by adding the following to smb.conf:

```
[tmp]
comment = temporary files
path = /tmp
read only = yes
```

THESE TESTS ASSUME VERSION 2.0.6 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 you IP name resolution is correctly set up. eg: 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"

Tests

Test 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 Or in: /usr/local/samba/lib

Test 2

Run the command "ping BIGSERVER" from the PC and "ping ACLIENT" from the unix box. If you don't get a valid response then your TCP/IP software is not correctly installed.

Note that 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 I assume 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 ipfwadm program.)

Test 3

Run the command "smbclient -L BIGSERVER" on the unix box. You should get a list of available shares back.

If you get a 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 "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".

If you get a "session request failed" then the server refused the connection. If it says "Your server software is being unfriendly" then its 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 following smb.conf file entries:

```
hosts deny = ALL
hosts allow = xxx.xxx.xxx.xxx/yy
bind interfaces only = Yes
```

In the above, no allowance has been made for any session requests that will automatically translate to the loopback adaptor address 127.0.0.1. To solve this problem change these lines to:

```
hosts deny = ALL
hosts allow = xxx.xxx.xxx.xxx/yy 127.
```

Do NOT use the "bind interfaces only" parameter where you may wish to use the samba password change

SAMBA Project Documentation

facility, or where smbclient may need to access local service for name resolution or for local resource connections. (Note: the "bind interfaces only" parameter deficiency where it will not allow connections to the loopback address will be fixed soon).

Another common cause of these two errors is having something already running on port 139, such as Samba (ie: 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 TEST 3 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.nmb file.

Test 4

Run the command "nmblookup -B BIGSERVER __SAMBA__". You should get the IP address of your Samba server back.

If you don't 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.

Test 5

run the command **nmblookup -B ACLIENT '*'**

You should get the PCs IP address back. If you don't 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 doesn't resolve via DNS then use the IP address of the client in the above test.

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/TCPIP hosts on the network should respond, although Samba may not catch all of the responses in the short time it listens. You should see "got a positive name query response" messages from several hosts.

If this doesn't 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 use 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 the 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).

Test 7

Run the command **smbclient //BIGSERVER/TMP**. You should then be prompted for a password. You should use the password of the account you are logged into the unix box with. If you want to test with another account then add the **-U >accountname<** option to the end of the command line. eg: **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 don't 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:

1. you have shadow passwords (or some other password system) but didn't compile in support for them in smbd
2. your "valid users" configuration is incorrect
3. you have a mixed case password and you haven't enabled the "password level" option at a high enough level
4. the "path =" line in smb.conf is incorrect. Check it with testparm
5. you enabled password encryption but didn't create the SMB encrypted password file

Once connected you should be able to use the commands **dir get put** etc. Type **help >command<** for instructions. You should especially check that the amount of free disk space shown is correct when you type **dir**.

Test 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 available shares on the server.

If you get a "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):

1. fixup the nmbd installation
2. add the IP address of BIGSERVER to the "wins server" box in the advanced tcp/ip setup on the PC.
3. enable windows name resolution via DNS in the advanced section of the tcp/ip setup
4. add BIGSERVER to your lmhosts file on the PC.

If you get a "invalid network name" or "bad password error" then the same fixes apply as they did 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 "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, etc.)

Test 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.

Test 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 don't 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.

Test 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 a "invalid password" error when you do then you are probably running WinNT 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 enable encrypted passwords AFTER compiling in support for encrypted passwords (refer to the Makefile).

Still having troubles?

Try the mailing list or newsgroup, or use the ethereal utility to sniff the problem. The official samba mailing list can be reached at samba@samba.org. To find out more about samba and how to subscribe to the mailing list check out the samba web page at <http://samba.org/samba>

Also look at the other docs in the Samba package!

Integrating MS Windows networks with Samba

Agenda

To identify the key functional mechanisms of MS Windows networking to enable the deployment of Samba as a means of extending and/or replacing MS Windows NT/2000 technology.

We will examine:

1. Name resolution in a pure Unix/Linux TCP/IP environment
2. Name resolution as used within MS Windows networking
3. How browsing functions and how to deploy stable and dependable browsing using Samba
4. MS Windows security options and how to configure Samba for seamless integration
5. Configuration of Samba as:
 - a. A stand-alone server
 - b. An MS Windows NT 3.x/4.0 security domain member
 - c. An alternative to an MS Windows NT 3.x/4.0 Domain Controller

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`

`/etc/hosts`

Contains a static list of IP Addresses and names. eg:

```
127.0.0.1    localhost localhost.localdomain
192.168.1.1  bigbox.caldera.com  bigbox  alias4box
```

The purpose of `/etc/hosts` is to provide a name resolution mechanism so that 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). eg: 168.192.1.1

MAC Addresses use 48 bits (or 6 bytes) and are typically represented as two digit hexadecimal numbers separated by colons. eg: 40:8e:0a:12:34:56

Every network interface must have a MAC address. Associated with a MAC address there 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 takes place using

SAMBA Project Documentation

MAC addressing. Since MAC addresses must be globally unique, and generally remains 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 UDP (User Datagram Protocol) to send a request to all interfaces on the local network segment using the all 1's 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 that a basic level of name resolution can exist before any other method of name resolution becomes available.

`/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

`/etc/host.conf`

`/etc/host.conf` is the primary means by which the setting in `/etc/resolv.conf` may be affected. 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.

`/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.
```

SAMBA Project Documentation

```
#

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+ hesoid 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 assumes a principal of speaking only when necessary.

Samba version 2.2.0 will add Linux support for extensions to the name service switch infrastructure so that 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 (ie: **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 it's NetBIOS machine name, so long as that machine is within the workgroup to which both the samba machine and the MS Windows machine belong.

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", "networking name", "netbios name", "SMB name". All terms mean the same thing with the exception of "netbios name" which can apply also to the name of the workgroup or the domain name. The terms "workgroup" and "domain" are really just a simply 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.

The following are typical NetBIOS name/service type registrations:

```
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

Group Names:
    WORKGROUP<03> = Generic Name registered by all members of WORKGROUP
    WORKGROUP<1c> = Domain Controllers / Netlogon Servers
    WORKGROUP<1d> = Local Master Browsers
```

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 find 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 `*<lc>`. A logon request is then sent to each IP address that is returned in the enumerated list of IP addresses. Which ever machine first replies 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 use of just a password (known as SHARE MODE 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 MODE security in a WORKGROUP environment, thus requiring 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.

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 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. i.e.: Its name is in the cache, so a name resolution lookup will succeed, but the machine can not respond. This can be frustrating for users – but it 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".

The LMHOSTS file

This file is usually located in MS Windows NT 4.0 or 2000 in 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 oriented.

It typically looks like:

```
# 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 comptrername
# 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
#     \0xnn (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 affects 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
```

SAMBA Project Documentation

```
# \0xnn 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.
```

HOSTS file

This file is usually located in MS Windows NT 4.0 or 2000 in 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.

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 what the NetBIOS Node Type parameter is configured to. A Node Type of 0 means use NetBIOS broadcast (over UDP broadcast) is first 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.

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.

How browsing functions and how to deploy stable and dependable browsing using Samba

As stated above, MS Windows machines register their NetBIOS names (i.e.: the machine name for each service type in operation) on start up. Also, as stated above, 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 11–15 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 synchronisation 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 synchronisation. 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 synchronisation – 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.

MS Windows security options and how to configure Samba for seamless integration

MS Windows clients may use encrypted passwords as part of a challenge/response authentication model (a.k.a. NTLMv1) or alone, or clear text 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 requests.

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.

You should refer to the [Password Encryption](#) chapter in this HOWTO collection for more details on the inner workings

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, they dropped support 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 mapping attempts to revalidate it will fail if the remote authentication server does not support encrypted passwords. This means that 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 client upper casing usernames and password before transmitting them to the SMB server when using clear text authentication.

```
password_level = integer
username_level = integer
```

By default Samba will 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 character, the *username_level* parameter is rarely even needed.

However, password on UNIX systems often make use of mixed case characters. This means that in order for a user on a Windows 9x client to connect to a Samba server using clear text authentication, the *password_level* must be set to the maximum number of upper case letter which *could* appear in a password. Note that if the server OS uses the traditional DES version of crypt(), then 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 where ever Samba is used. There are three configuration possibilities for support of encrypted passwords:

Use 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_PDC"
```

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

The down-side 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 does require there to be a standard Unix account for the user, this account can be blocked to prevent logons by other than MS Windows clients.

Make Samba a member of an MS Windows NT security domain

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

```
encrypt passwords = Yes
security = domain
workgroup = "name of NT domain"
password server = *
```

The use of the "*" argument to "password server" will cause samba to locate the domain controller in a way analogous to the way this is done within MS Windows NT.

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

- On the MS Windows NT domain controller using the Server Manager add a machine account for the Samba server.
- Next, on the Linux system execute: **smbpasswd -r PDC_NAME -j DOMAIN_NAME**

Use of this mode of authentication does require there to be a standard Unix account for the 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 other than MS Windows clients by things 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 the [Winbind Overview](#) chapter in this HOWTO collection.

Configure Samba as an authentication server

This mode of authentication demands that there be on the Unix/Linux system both a Unix style account as well as an smbpasswd entry for the user. The Unix system account can be locked if required as only the encrypted password will be used for SMB client authentication.

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

```
## please refer to the Samba PDC HOWTO chapter later in
## this collection for more details
[global]
    encrypt passwords = Yes
    security = user
    domain logons = Yes
    ; an OS level of 33 or more is recommended
    os level = 33

[NETLOGON]
    path = /somewhere/in/file/system
    read only = yes
```

in order for this method to work a Unix system account needs to be created for each user, as well as for each MS Windows NT/2000 machine. The following structure is required.

Users

A user account that may provide a home directory should be created. The following Linux system commands are typical of the procedure for creating an account.

```
# useradd -s /bin/bash -d /home/"userid" -m "userid"
# passwd "userid"
Enter Password: <pw>

# smbpasswd -a "userid"
Enter Password: <pw>
```

MS Windows NT Machine Accounts

These are required only when Samba is used as a domain controller. Refer to the Samba-PDC-HOWTO for more details.

```
# useradd -s /bin/false -d /dev/null "machine_name"\$
# passwd -l "machine_name"\$
# smbpasswd -a -m "machine_name"
```

Conclusions

Samba provides a flexible means to operate as...

- A Stand-alone server – No special action is needed other than to create user accounts. Stand-alone servers do NOT provide network logon services, meaning that machines that use this server do NOT perform a domain logon but instead make use only of the MS Windows logon which is local to the MS Windows workstation/server.

SAMBA Project Documentation

- An MS Windows NT 3.x/4.0 security domain member.
 - An alternative to an MS Windows NT 3.x/4.0 Domain Controller.
-

Configuring PAM for distributed but centrally managed authentication

Samba and PAM

A number of Unix systems (eg: 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 such programs as: **login**, **passwd**, **chown**, etc.

PAM provides a mechanism that disconnects these security programs from the underlying authentication/authorization infrastructure. PAM is configured either through one file `/etc/pam.conf` (Solaris), or by editing individual files that are located in `/etc/pam.d`.

The following is an example `/etc/pam.d/login` configuration file. This example had all options been uncommented is probably not usable as 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`.

```
##%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
```

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_mkhomedir.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
```

SAMBA Project Documentation

The following example for the login program replaces the use of the `pam_pwdb.so` module which 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      /lib/security/pam_pwdb.so nullok nodelay shadow audit
account     required      /lib/security/pam_pwdb.so audit nodelay
session     required      /lib/security/pam_pwdb.so nodelay
password    required      /lib/security/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      /lib/security/pam_smbpass.so nodelay
account     required      /lib/security/pam_pwdb.so audit nodelay
session     required      /lib/security/pam_pwdb.so nodelay
password    required      /lib/security/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 very 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 examine the PAM documentation for further helpful information.

Distributed Authentication

The astute administrator will realize from this that the combination of `pam_smbpass.so`, `winbindd`, and `rsync` (see <http://rsync.samba.org/>) will allow the establishment of a centrally managed, distributed

user/password database that can also be used by all PAM (eg: Linux) aware programs and applications. This arrangement can have particularly potent advantages compared with the use of Microsoft Active Directory Service (ADS) in so far as reduction of wide area network authentication traffic.

PAM Configuration in smb.conf

There is an option in smb.conf called obey pam restrictions. The following is from the on-line help for this option in SWAT;

When Samba 2.2 is configure 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**

Hosting a Microsoft Distributed File System tree on Samba

Instructions

The Distributed File System (or 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 etc. For more information about Dfs, refer to [Microsoft documentation](#).

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 2000.

Here's an example of setting up a Dfs tree on a Samba server.

```
# The smb.conf file:
[global]
    netbios name = SAMBA
    host msdfs = yes

[dfs]
    path = /export/dfsroot
    msdfs root = yes
```

In the `/export/dfsroot` directory we set up our 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
```

You should set up the permissions and ownership of the directory acting as the Dfs root such 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.

SAMBA Project Documentation

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.

Notes

- 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 that only designated users can modify the symbolic links in the directory.
-

UNIX Permission Bits and Windows NT Access Control Lists

Viewing and changing UNIX permissions using the NT security dialogs

New in the Samba 2.0.4 release is the ability for Windows NT clients to use their native security settings dialog box to view and modify the underlying UNIX permissions.

Note that this ability is careful not to compromise the security of the UNIX host Samba is running on, and still obeys all the file permission rules that a Samba administrator can set.

In Samba 2.0.4 and above the default value of the parameter `nt_acl_support` has been changed from `false` to `true`, so manipulation of permissions is turned on by default.

How to view file security on a Samba share

From an NT 4.0 client, single-click with the right mouse button 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 normal file properties dialog box, but with Samba 2.0.4 this will have a new tab along the top marked *Security*. Click on this 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.

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 of the form :

"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). Click on the **Close** button to remove this dialog.

If the parameter `nt_acl_support` is set to `false` then 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 on 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 2.0.4 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, available from the main Samba ftp site.

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 in the form :

"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* then 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.

File Permissions

The standard UNIX user/group/world triple and the corresponding "read", "write", "execute" permissions triples 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.

As many UNIX permission sets don't map into common NT names such as "**read**", "**change**" or "**full control**" then 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 then Samba 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 will be given below.

Directory Permissions

Directories on an NT NTFS file system have two different sets of permissions. The first set of permissions is the ACL set on the directory itself, this 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.

Modifying file or directory permissions

Modifying file and directory permissions is as simple as changing the displayed permissions in the dialog box, and clicking the **OK** button. 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` then 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 2.0.4 (it will give an error message of **"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 triple (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 triple component.

As UNIX supports only the "r", "w" and "x" bits of an NT ACL then if other NT security attributes such as "Delete access" are selected then 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 **"OK"**.

If you wish to remove all permissions from a user/group/world component then you may either highlight the component and click the **"Remove"** button, or set the component to only have the special **"Take Ownership"** permission (displayed as **"O"**) highlighted.

Interaction with the standard Samba create mask parameters

Note that with Samba 2.0.5 there are four new parameters to control this interaction. These are :

security mask

force security mode

directory security mask

force directory security mode

SAMBA Project Documentation

Once a user clicks "**OK**" to apply the permissions Samba maps the given permissions into a user/group/world r/w/x triple set, and then will check 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* 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 set explicitly this parameter is set to the same value as the *create mask* parameter to provide compatibility with Samba 2.0.4 where this permission change facility was introduced. 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 set explicitly this parameter is set to the same value as the *force create mode* parameter to provide compatibility with Samba 2.0.4 where the permission change facility was introduced. 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 using 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 to provide compatibility with Samba 2.0.4 where the permission change facility was introduced.

In this way Samba enforces the permission restrictions that an administrator can set on a Samba share, whilst 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 doesn't force any particular bits to be set 'on', then set the following parameters in the *smb.conf(5)* file in that share specific section :

```
security mask = 0777
```

```
force security mode = 0
```

```
directory security mask = 0777
```

```
force directory security mode = 0
```

As described, in Samba 2.0.4 the parameters :

create mask

force create mode

directory mask

force directory mode

were used instead of the parameters discussed here.

Interaction with the standard Samba file attribute mapping

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.

One way this can show up is 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 info in another tab.

What this can mean is that if the owner changes the permissions to allow themselves read access using the security dialog, clicks **"OK"** to get back to the standard attributes tab dialog, and then clicks **"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 **"OK"** to get back to the attributes dialog you should always hit **"Cancel"** rather than **"OK"** to ensure that your changes are not overridden.

Printing Support in Samba 2.2.x

Introduction

Beginning with the 2.2.0 release, Samba supports the native Windows NT printing mechanisms implemented via MS-RPC (i.e. the SPOOLSS named pipe). Previous versions of Samba only supported LanMan printing calls.

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.
- Uploading of printer drivers via the Windows NT Add Printer Wizard (APW) or the Imprints tool set (refer to <http://imprints.sourceforge.net>).
- Support for the native MS-RPC printing calls such as StartDocPrinter, EnumJobs(), etc... (See the MSDN documentation at <http://msdn.microsoft.com/> 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 an internal databases for spooled job information

There has been some initial confusion about what all this means and whether or not it is a requirement for printer drivers to be installed on a Samba host in order to support printing from Windows clients. A bug existed in Samba 2.2.0 which made Windows NT/2000 clients require that the Samba server possess a valid driver for the printer. This is fixed in Samba 2.2.1 and once again, Windows NT/2000 clients can use the local APW for installing drivers to be used with a Samba served printer. This is the same behavior exhibited by Windows 9x clients. As a side note, Samba does not use these drivers in any way to process spooled files. They are utilized entirely by the clients.

The following MS KB article, may be of some help if you are dealing with Windows 2000 clients: *How to Add Printers with No User Interaction in Windows 2000*

<http://support.microsoft.com/support/kb/articles/Q189/1/05.ASP>

Configuration

[print\$] vs. [printer\$]

Previous versions of Samba recommended using a share named [printer\$]. This name was taken from the printer\$ service created by Windows 9x clients when a printer was shared. Windows 9x printer servers always have a printer\$ service which provides read-only access via no password in order to support printer driver downloads.

However, the initial implementation allowed for a parameter named *printer driver location* to be used on a per share basis to specify 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 *printer driver file* parameter, are being deprecated and should not be used in new installations. For more information on this change, you should refer to the [Migration](#)

section of this document.

Creating [print\$]

In order to support the uploading of printer driver files, you must first configure a file share named [print\$]. The name of this share is hard coded in Samba's internals so the name is very important (print\$ is the service used by Windows NT print servers to provide support for printer driver download).

You should modify the server's smb.conf file to add the global parameters and to create the following file share (of course, some of the parameter values, such as 'path' are arbitrary and should be replaced with appropriate values for your site):

```
[global]
; members of the ntadmin group should be able
; to add drivers and set printer properties
; root is implicitly a 'printer admin'
printer admin = @ntadmin

[print$]
path = /usr/local/samba/printers
guest ok = yes
browseable = yes
read only = yes
; since this share is configured as read only, then we need
; a 'write list'. Check the file system permissions to make
; sure this account can copy files to the share. If this
; is setup to a non-root account, then it should also exist
; as a 'printer admin'
write list = @ntadmin,root
```

The write list is used to allow administrative level user accounts to have write access in order to update files on the share. See the [smb.conf\(5\) man page](#) for more information on configuring file shares.

The requirement for guest ok = yes depends upon how your site is configured. If users will be guaranteed to have an account on the Samba host, then this is a non-issue.

Author's Note

The non-issue is that if all your Windows NT users are guaranteed to be authenticated by the Samba server (such as a domain member server and the NT user has already been validated by the Domain Controller in order to logon to the Windows NT console), then guest access is not necessary. Of course, in a workgroup environment where you just want to be able to print without worrying about silly accounts and security, then configure the share for guest access. You'll probably want to add map to guest = Bad User in the [global] section as well. Make sure you understand what this parameter does before using it though. —jerry

In order for a Windows NT print server to support the downloading of driver files by multiple client architectures, it must create subdirectories within the [print\$] service which correspond to each of the supported client architectures. Samba follows this model as well.

Next create the directory tree below the [print\$] share for each architecture you wish to support.

```
[print$]-----
|-W32X86           ; "Windows NT x86"
```

```
| -WIN40           ; "Windows 95/98"
| -W32ALPHA       ; "Windows NT Alpha_AXP"
| -W32MIPS        ; "Windows NT R4000"
| -W32PPC        ; "Windows NT PowerPC"
```

ATTENTION! REQUIRED PERMISSIONS

In order to currently add a new driver to you 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 a member of the *printer admin* list.

Of course, the connected account must still possess 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, simply log onto the Samba server using a root (or *printer admin*) account from a Windows NT 4.0/2k client. Open "Network Neighbourhood" or "My Network Places" and browse for the Samba host. Once you have located the server, navigate to the "Printers..." folder. You should see an initial listing of printers that matches the printer shares defined on your Samba host.

Setting Drivers for Existing Printers

The initial listing of printers in the Samba host's Printers folder will have no real printer driver assigned to them. By default, in Samba 2.2.0 this driver name was set to *NO PRINTER DRIVER AVAILABLE FOR THIS PRINTER*. Later versions changed this to a NULL string to allow the use of the local Add Printer Wizard on NT/2000 clients. Attempting to view the printer properties for a printer which has this default driver assigned will result in the 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?

Click "No" in the error dialog and you will be presented with the printer properties window. The way to assign a driver to a printer is to either

- Use the "New Driver..." button to install a new printer driver, or
- Select a driver from the popup list of installed drivers. Initially this list will be empty.

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 a root account, you will also be able modify other printer properties such as ACLs and device settings using this dialog box.

A few closing comments for this section, it is possible on a Windows NT print server to have printers listed in the Printers folder which are not shared. Samba does not make this distinction. By definition, the only printers of which Samba is aware are those which are specified as shares in `smb.conf`.

Another interesting side note is that Windows NT clients do not use the SMB printer share, but rather can print directly to any printer on another Windows NT host using MS-RPC. This of course assumes that the printing client has the necessary privileges on the remote host serving the printer. The default permissions

assigned by Windows NT to a printer gives the "Print" permissions to the "Everyone" well-known group.

Support a large number of printers

One issue that has arisen during the development phase of Samba 2.2 is the need to support driver downloads for 100's of printers. Using the Windows NT APW is somewhat awkward to say the least. If more than one printer are using the same driver, the rpcclient's setdriver command can be used to set the driver associated with an installed driver. The following is an example of how this could be accomplished:

```
$ rpcclient pogo -U root%secret -c "enumdrivers"
Domain=[NARNIA] OS=[Unix] Server=[Samba 2.2.0-alpha3]

[Windows NT x86]
Printer Driver Info 1:
  Driver Name: [HP LaserJet 4000 Series PS]

Printer Driver Info 1:
  Driver Name: [HP LaserJet 2100 Series PS]

Printer Driver Info 1:
  Driver Name: [HP LaserJet 4Si/4SiMX PS]

$ rpcclient pogo -U root%secret -c "enumprinters"
Domain=[NARNIA] OS=[Unix] Server=[Samba 2.2.0-alpha3]
  flags:[0x800000]
  name:[\\POGO\hp-print]
  description:[POGO\\POGO\hp-print,NO DRIVER AVAILABLE FOR THIS PRINTER,]
  comment:[]

$ rpcclient pogo -U root%secret \
> -c "setdriver hp-print \"HP LaserJet 4000 Series PS\""
Domain=[NARNIA] OS=[Unix] Server=[Samba 2.2.0-alpha3]
Successfully set hp-print to driver HP LaserJet 4000 Series PS.
```

Adding New Printers via the Windows NT APW

By default, Samba offers all printer shares defined in `smb.conf` in the "Printers..." folder. Also existing 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`).
- `show add printer wizard = yes` (the default).

In order to be able 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 system (i.e. `/etc/printcap` or appropriate files) and `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 `smb.conf` 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. Note that the `add printer program` is executed under the context of the connected user, not necessarily a root account.

There is a complementary `delete printer command` for removing entries from the "Printers..." folder.

SAMBA Project Documentation

The following is an example [add printer command](#) script. It adds the appropriate entries to `/etc/printcap.local` (change that to what you need) and returns a line of 'Done' which is needed for the whole process to work.

```
#!/bin/sh

# Script to insert a new printer entry into printcap.local
#
# $1, printer name, used as the descriptive name
# $2, share name, used as the printer name for Linux
# $3, port name
# $4, driver name
# $5, location, used for the device file of the printer
# $6, win9x location

#
# Make sure we use the location that RedHat uses for local printer defs
PRINTCAP=/etc/printcap.local
DATE=`date +%Y%m%d-%H%M%S`
LP=lp
RESTART="service lpd restart"

# Keep a copy
cp $PRINTCAP $PRINTCAP.$DATE
# Add the printer to $PRINTCAP
echo "" >> $PRINTCAP
echo "$2|$1:\\\\" >> $PRINTCAP
echo " :sd=/var/spool/lpd/$2:\\\\" >> $PRINTCAP
echo " :mx=0:ml=0:sh:\\\\" >> $PRINTCAP
echo " :lp=/usr/local/samba/var/print/$5.prn:" >> $PRINTCAP

touch "/usr/local/samba/var/print/$5.prn" >> /tmp/printadd.$$ 2>&1
chown $LP "/usr/local/samba/var/print/$5.prn" >> /tmp/printadd.$$ 2>&1

mkdir /var/spool/lpd/$2
chmod 700 /var/spool/lpd/$2
chown $LP /var/spool/lpd/$2
#echo $1 >> "/usr/local/samba/var/print/$5.prn"
#echo $2 >> "/usr/local/samba/var/print/$5.prn"
#echo $3 >> "/usr/local/samba/var/print/$5.prn"
#echo $4 >> "/usr/local/samba/var/print/$5.prn"
#echo $5 >> "/usr/local/samba/var/print/$5.prn"
#echo $6 >> "/usr/local/samba/var/print/$5.prn"
$RESTART >> "/usr/local/samba/var/print/$5.prn"
# Not sure if this is needed
touch /usr/local/samba/lib/smb.conf
#
# You need to return a value, but I am not sure what it means.
#
echo "Done"
exit 0
```

Samba and Printer Ports

Windows NT/2000 print servers associate a port with each printer. These normally take the form of LPT1:, COM1:, FILE:, etc... 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 a port in order to print, rather it is a requirement of Windows clients.

Note that Samba does not support the concept of "Printer Pooling" internally either. This is when a logical printer is assigned to multiple ports as a form of load balancing or fail over.

If you require that multiple ports be defined for some reason, `smb.conf` possesses a *[enumports command](#)* which can be used to define an external program that generates a listing of ports on a system.

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 will only provide a brief introduction to the features of Imprints.

What is Imprints?

Imprints is a collection of tools for supporting the goals of

- Providing a central repository 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 which will obtain and install printer drivers on remote Samba and Windows NT 4 print servers.
-

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.

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 package downloaded is actually the one referred in the Imprints database. It is *not* recommended that this security check be disabled.

The Installation Client

More information regarding the Imprints installation client is available in the `Imprints-Client-HOWTO.ps` file 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.

SAMBA Project Documentation

The basic installation process is in four steps and perl code is wrapped around **smbclient** and **rpcclient**.

```
foreach (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
}
4. 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. As 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 ok as Windows NT always requires that at least the Windows NT version of the printer driver is present. However, 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 NT driver is installed first.

Migration to from Samba 2.0.x to 2.2.x

Given that printer driver management has changed (we hope improved) in 2.2 over prior releases, migration from an existing setup to 2.2 can follow several paths. Here are the possible scenarios for migration:

- If you do not desire the new Windows NT print driver support, nothing needs to be done. All existing parameters work the same.
- If you want to take advantage of NT printer driver support but do not want to migrate the 9x drivers to the new setup, the leave the existing `printers.def` file. When `smbd` attempts to locate a 9x driver for the printer in the TDB and fails it will drop down to using the `printers.def` (and all associated parameters). The **make_printerdef** tool will also remain for backwards compatibility but will be removed in the next major release.
- If you install a Windows 9x driver for a printer on your Samba host (in the printing TDB), this information will take precedence and the three old printing parameters will be ignored (including print driver location).
- If you want to migrate an existing `printers.def` file into the new setup, the current only solution is to use the Windows NT APW to install the NT drivers and the 9x drivers. This can be scripted using **smbclient** and **rpcclient**. See the Imprints installation client at <http://imprints.sourceforge.net/> for an example.

Achtung!

SAMBA Project Documentation

The following `smb.conf` parameters are considered to be deprecated and will be removed soon. Do not use them in new installations

- *printer driver file (G)*
- *printer driver (S)*
- *printer driver location (S)*

There have been two new parameters added in Samba 2.2.2 to for better support of Samba 2.0.x backwards capability (*disable spoolss*) and for using local printers drivers on Windows NT/2000 clients (*use client driver*). Both of these options are described in the `smb.conf(5)` man page and are disabled by default.

Debugging Printing Problems

Introduction

This is a short description of how to debug printing problems with Samba. This describes how to debug problems with printing from a SMB client to a Samba server, not the other way around. For the reverse see the examples/printing directory.

Ok, so you want to print to a Samba server from your PC. The first thing you need to understand is that Samba does not actually do any printing itself, it just acts as a middleman between your PC client and your Unix printing subsystem. Samba receives the file from the PC then passes the file to a external "print command". What print command you use is up to you.

The whole things is controlled using options in smb.conf. The most relevant options (which you should look up in the smb.conf man page) are:

```
[global]
  print command      - send a file to a spooler
  lpq command        - get spool queue status
  lprm command       - remove a job
[printers]
  path = /var/spool/lpd/samba
```

The following are nice to know about:

```
queuepause command  - stop a printer or print queue
queueresume command - start a printer or print queue
```

Example:

```
print command = /usr/bin/lpr -r -P%p %s
lpq command   = /usr/bin/lpq   -P%p %s
lprm command  = /usr/bin/lprm  -P%p %j
queuepause command = /usr/sbin/lpc -P%p stop
queueresume command = /usr/sbin/lpc -P%p start
```

Samba should set reasonable defaults for these depending on your system type, but it isn't clairvoyant. It is not uncommon that you have to tweak these for local conditions. The commands should always have fully specified pathnames, as the smdb may not have the correct PATH values.

When you send a job to Samba to be printed, it will make a temporary copy of it in the directory specified in the [printers] section. and it should be periodically cleaned out. The lpr -r option requests that the temporary copy be removed after printing; If printing fails then you might find leftover files in this directory, and it should be periodically cleaned out. Samba used the lpq command to determine the "job number" assigned to your print job by the spooler.

The %>letter< are "macros" that get dynamically replaced with appropriate values when they are used. The %s gets replaced with the name of the spool file that Samba creates and the %p gets replaced with the name of the printer. The %j gets replaced with the "job number" which comes from the lpq output.

Debugging printer problems

One way to debug printing problems is to start by replacing these command with shell scripts that record the arguments and the contents of the print file. A simple example of this kind of things might be:

```
print command = /tmp/saveprint %p %s

#!/bin/saveprint
# we make sure that we are the right user
/usr/bin/id -p >/tmp/tmp.print
# we run the command and save the error messages
# replace the command with the one appropriate for your system
/usr/bin/lpr -r -P$1 $2 2>>&/tmp/tmp.print
```

Then you print a file and try removing it. You may find that the print queue needs to be stopped in order to see the queue status and remove the job:

```
h4: {42} % echo hi >/tmp/hi
h4: {43} % smbclient //localhost/lw4
added interface ip=10.0.0.4 bcast=10.0.0.255 nmask=255.255.255.0
Password:
Domain=[ASTART] OS=[Unix] Server=[Samba 2.0.7]
smb: \> print /tmp/hi
putting file /tmp/hi as hi-17534 (0.0 kb/s) (average 0.0 kb/s)
smb: \> queue
1049      3          hi-17534
smb: \> cancel 1049
Error cancelling job 1049 : code 0
smb: \> cancel 1049
Job 1049 cancelled
smb: \> queue
smb: \> exit
```

The 'code 0' indicates that the job was removed. The comment by the smbclient is a bit misleading on this. You can observe the command output and then look at the /tmp/tmp.print file to see what the results are. You can quickly find out if the problem is with your printing system. Often people have problems with their /etc/printcap file or permissions on various print queues.

What printers do I have?

You can use the 'testprns' program to check to see if the printer name you are using is recognized by Samba. For example, you can use:

```
testprns printer /etc/printcap
```

Samba can get its printcap information from a file or from a program. You can try the following to see the format of the extracted information:

```
testprns -a printer /etc/printcap

testprns -a printer '|/bin/cat printcap'
```

Setting up printcap and print servers

You may need to set up some printcaps for your Samba system to use. It is strongly recommended that you use the facilities provided by the print spooler to set up queues and printcap information.

Samba requires either a printcap or program to deliver printcap information. This printcap information has the format:

```
name|alias1|alias2...:option=value:...
```

For almost all printing systems, the printer 'name' must be composed only of alphanumeric or underscore '_' characters. Some systems also allow hyphens ('-') as well. An alias is an alternative name for the printer, and an alias with a space in it is used as a 'comment' about the printer. The printcap format optionally uses a \ at the end of lines to extend the printcap to multiple lines.

Here are some examples of printcap files:

1. pr just printer name
2. pr|alias printer name and alias
3. pr|My Printer printer name, alias used as comment
4. pr:sh:\ Same as pr:sh:cm= testing :cm= \ testing
5. pr:sh Same as pr:sh:cm= testing :cm= testing

Samba reads the printcap information when first started. If you make changes in the printcap information, then you must do the following:

1. make sure that the print spooler is aware of these changes. The LPRng system uses the 'lpc reread' command to do this.
2. make sure that the spool queues, etc., exist and have the correct permissions. The LPRng system uses the 'checkpc -f' command to do this.
3. You now should send a SIGHUP signal to the smbd server to have it reread the printcap information.

Job sent, no output

This is the most frustrating part of printing. You may have sent the job, verified that the job was forwarded, set up a wrapper around the command to send the file, but there was no output from the printer.

First, check to make sure that the job REALLY is getting to the right print queue. If you are using a BSD or LPRng print spooler, you can temporarily stop the printing of jobs. Jobs can still be submitted, but they will not be printed. Use:

```
lpc -Pprinter stop
```

Now submit a print job and then use 'lpq -Pprinter' to see if the job is in the print queue. If it is not in the print queue then you will have to find out why it is not being accepted for printing.

Next, you may want to check to see what the format of the job really was. With the assistance of the system administrator you can view the submitted jobs files. You may be surprised to find that these are not in what you would expect to call a printable format. You can use the UNIX 'file' utility to determine what the job

format actually is:

```
cd /var/spool/lpd/printer # spool directory of print jobs
ls # find job files
file dfA001myhost
```

You should make sure that your printer supports this format OR that your system administrator has installed a 'print filter' that will convert the file to a format appropriate for your printer.

Job sent, strange output

Once you have the job printing, you can then start worrying about making it print nicely.

The most common problem is extra pages of output: banner pages OR blank pages at the end.

If you are getting banner pages, check and make sure that the printcap option or printer option is configured for no banners. If you have a printcap, this is the :sh (suppress header or banner page) option. You should have the following in your printer.

```
printer: ... :sh
```

If you have this option and are still getting banner pages, there is a strong chance that your printer is generating them for you automatically. You should make sure that banner printing is disabled for the printer. This usually requires using the printer setup software or procedures supplied by the printer manufacturer.

If you get an extra page of output, this could be due to problems with your job format, or if you are generating PostScript jobs, incorrect setting on your printer driver on the MicroSoft client. For example, under Win95 there is a option:

```
Printers|Printer Name|(Right Click)Properties|Postscript|Advanced|
```

that allows you to choose if a Ctrl-D is appended to all jobs. This is a very bad thing to do, as most spooling systems will automatically add a ^D to the end of the job if it is detected as PostScript. The multiple ^D may cause an additional page of output.

Raw PostScript printed

This is a problem that is usually caused by either the print spooling system putting information at the start of the print job that makes the printer think the job is a text file, or your printer simply does not support PostScript. You may need to enable 'Automatic Format Detection' on your printer.

Advanced Printing

Note that you can do some pretty magic things by using your imagination with the "print command" option and some shell scripts. Doing print accounting is easy by passing the %U option to a print command shell script. You could even make the print command detect the type of output and its size and send it to an appropriate printer.

Real debugging

If the above debug tips don't help, then maybe you need to bring in the bug guns, system tracing. See `Tracing.txt` in this directory.

Security levels

Introduction

Samba supports the following options to the global smb.conf parameter

```
[global]
security = [share|user(default)|domain|ads]
```

Please refer to the smb.conf man page for usage information and to the document [DOMAIN_MEMBER.html](#) for further background details on domain mode security. The Windows 2000 Kerberos domain security model (security = ads) is described in the [ADS-HOWTO.html](#).

Of the above, "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. That real authentication server can be another Samba server or can be a Windows NT server, the later natively capable of encrypted password support.

More complete description of security levels

A 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. I know this is 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.

I'll describe user level security first, as its simpler. In user level security the client will send a "session setup" command directly after the protocol negotiation. This contains a username and password. The server can either accept or reject that username/password combination. Note that 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 machine that the client is coming from

If the server accepts the username/password then the client expects to be able to mount any share (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)

Ok, now for share level security. In share level security the client authenticates itself separately for each share. It will send a password along with each "tree connection" (share mount). It does not explicitly send a username with this operation. The client is expecting 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

SAMBA Project Documentation

where it is a username/password that is authenticated, not a "share/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 =" smb.conf line. The password is then checked in turn against these "possible usernames". If a match is found then the client is authenticated as that user.

Finally "server level" security. In server level security 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 clients connection. This allows the samba server to use another SMB server as the "password server".

You should also note that at the very 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 then it supplies the client with a random "cryptkey". The client will then send all passwords in encrypted form. You have to compile samba with encryption enabled to support this feature, and you have to maintain a separate smbpasswd file with SMB style encrypted passwords. It is cryptographically impossible to translate from unix style encryption to SMB style encryption, although there are some fairly simple management schemes by which the two could be kept in sync.

security = domain in Samba 2.x

Joining an NT Domain with Samba 2.2

Assume you have a Samba 2.x server with a NetBIOS name of `SERV1` and are joining an NT domain called `DOM`, which has a PDC with a NetBIOS name of `DOMPDC` and two backup domain controllers with NetBIOS names `DOMBDC1` and `DOMBDC2`.

In order to join the domain, first stop all Samba daemons and run the command:

```
root# smbpasswd -j DOM -r DOMPDC -UAdministrator%password
```

as we are joining the domain `DOM` and the PDC for that domain (the only machine that has write access to the domain SAM database) is `DOMPDC`. The `Administrator%password` is the login name and password for an account which has the necessary privilege to add machines to the domain. If this is successful you will see the message:

```
smbpasswd: Joined domain DOM.
```

in your terminal window. See the [smbpasswd\(8\)](#) man page for more details.

There is existing development code to join a domain without having to create the machine trust account on the PDC beforehand. This code will hopefully be available soon in release branches as well.

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 an `smbpasswd` file would be stored – normally :

```
/usr/local/samba/private
```

In Samba 2.0.x, the filename looks like this:

```
<NT DOMAIN NAME>.<Samba Server Name>.mac
```

The `.mac` suffix stands for machine account password file. So in our example above, the file would be called:

```
DOM.SERV1.mac
```

In Samba 2.2, this file has been replaced with a TDB (Trivial Database) file named `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.

Now, before restarting the Samba daemons you must edit your [smb.conf\(5\)](#) 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 = DOM

as 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.

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.

Alternatively, 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, which was introduced in Samba 2.0.6, allows Samba to use exactly the same mechanism that NT does. This method either broadcasts or uses a WINS database in order to find domain controllers to authenticate against.

Finally, restart your Samba daemons and get ready for clients to begin using domain security!

Samba and Windows 2000 Domains

Many people have asked regarding the state of Samba's ability to participate in a Windows 2000 Domain. Samba 2.2 is able to act as a member server of a Windows 2000 domain operating in mixed or native mode.

There is much confusion between the circumstances that require a "mixed" mode Win2k DC and a when this host can be switched to "native" mode. A "mixed" mode Win2k domain controller is only needed if Windows NT BDCs must exist in the same domain. By default, a Win2k DC in "native" mode will still support NetBIOS and NTLMv1 for authentication of legacy clients such as Windows 9x and NT 4.0. Samba has the same requirements as a Windows NT 4.0 member server.

The steps for adding a Samba 2.2 host to a Win2k domain are the same as those for adding a Samba server to a Windows NT 4.0 domain. The only exception is that the "Server Manager" from NT 4 has been replaced by the "Active Directory Users and Computers" MMC (Microsoft Management Console) plugin.

Why is this better than `security = server`?

Currently, domain security in Samba doesn't 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 filesystem. This is very similar to the older Samba security mode `security = server`, where Samba would pass through the

SAMBA Project Documentation

authentication request to a Windows NT server in the same way as a Windows 95 or Windows 98 server would.

Please refer to the [Winbind paper](#) for information on a system to automatically assign UNIX uids and gids to Windows NT Domain users and groups. This code is available in development branches only at the moment, but will be moved to release branches soon.

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, etc. All this information will allow Samba to be extended in the future into a mode the developers currently call appliance mode. In this mode, no local Unix users will be necessary, and Samba will generate Unix uids and gids from the information passed back from the PDC when a user is authenticated, making a Samba server truly plug and play in an NT domain environment. Watch for this code soon.

NOTE: Much of the text of this document was first published in the Web magazine [LinuxWorld](#) as the article [Doing the NIS/NT Samba](#).

Unified Logons between Windows NT and UNIX using Winbind

Abstract

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. We present *winbind*, a component of the Samba suite of programs as a solution to 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 paper describes the winbind system, explaining the functionality it provides, how it is configured, and how it works internally.

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.

What Winbind Provides

Winbind unifies UNIX and Windows NT account management by allowing a UNIX box to become a full member of a 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.

SAMBA Project Documentation

Users on the UNIX machine can then use NT user and group names as they would use "native" UNIX names. They can chown files so that 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 a 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).

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 a 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.

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 processed sequentially.

The technologies used to implement winbind are described in detail below.

Microsoft Remote Procedure Calls

Over the last two 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 which 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.

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 a 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 which 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 species 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 a very 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.

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 for more details.

User and Group ID Allocation

When a user or group is created under Windows NT is it allocated a numerical relative identifier (RID). This is slightly different to 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.

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.

Installation and Configuration

Many thanks to John Trostel jtrostel@snapserver.com for providing the HOWTO for this section.

This HOWTO describes how to get winbind services up and running to control access and authenticate users on your Linux box using the winbind services which come with SAMBA 2.2.2.

There is also some Solaris specific information in `docs/textdocs/Solaris-Winbind-HOWTO.txt`. Future revisions of this document will incorporate that information.

Introduction

This HOWTO describes the procedures used to get winbind up and running on my RedHat 7.1 system. Winbind is capable of providing access and authentication control for Windows Domain users through an NT or Win2K PDC for 'regular' services, such as telnet and ftp, as well for SAMBA services.

This HOWTO has been written from a 'RedHat-centric' perspective, so if you are using another distribution, you may have to modify the instructions somewhat to fit the way your distribution works.

- *Why should I do this?*

SAMBA Project Documentation

This allows the SAMBA administrator to rely on the authentication mechanisms on the NT/Win2K PDC for the authentication of domain members. NT/Win2K users no longer need to have separate accounts on the SAMBA server.

- *Who should be reading this document?*

This HOWTO is designed for system administrators. If you are implementing SAMBA on a file server and wish to (fairly easily) integrate existing NT/Win2K users from your PDC onto the SAMBA server, this HOWTO is for you. That said, I am no NT or PAM expert, so you may find a better or easier way to accomplish these tasks.

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 (version 2.2.2 as of this writing), now includes a functioning winbindd daemon. Please refer to the [main SAMBA web page](#) 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 (pluggable authentication modules) must be setup properly on your machine. In order to compile the winbind modules, you should have at least the pam libraries resident on your system. For recent RedHat systems (7.1, for instance), that means `pam-0.74-22`. For best results, it is helpful to also install the development packages in `pam-devel-0.74-22`.

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, you will want to make sure that you have the standard PAM package (for RedHat) which supplies the /etc/pam.d directory structure, including the pam modules 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 was also installed. This package includes the header files needed to compile pam-aware applications. For instance, my RedHat system has both `pam-0.74-22` and `pam-devel-0.74-22` RPMs installed.

Configure and compile SAMBA

The configuration and compilation of SAMBA is pretty straightforward. The first three steps may not be necessary depending upon whether or not you have previously built the Samba binaries.

```
root# autoconf
root# make clean
root# rm config.cache
root# ./configure --with-winbind
root# make
root# make install
```

SAMBA Project Documentation

This will, by default, install SAMBA in `/usr/local/samba`. See the main SAMBA documentation if you want to install SAMBA somewhere else. It will also build the `winbindd` executable and libraries.

Configure `nsswitch.conf` and the winbind libraries

The libraries needed to run the `winbindd` daemon through `nsswitch` need to be copied to their proper locations, so

```
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 winbind daemon will be automatically entered into the `ldconfig` cache the next time your system reboots, but it is faster (and you don't 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.

Configure `smb.conf`

Several parameters are needed in the `smb.conf` file to control the behavior of `winbindd`. Configure `smb.conf` These are described in more detail in the [winbindd\(8\)](#) man page. My `smb.conf` file was modified to include the following entries in the `[global]` section:

```
[global]
<...>
# separate domain and username with '+', like DOMAIN+username
winbind_separator = +
# use uids from 10000 to 20000 for domain users
winbind_uid = 10000-20000
# use gids from 10000 to 20000 for domain groups
winbind_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
```

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.

Start up the winbindd daemon and test it!

Eventually, you will want to modify your smb startup script to automatically invoke the winbindd daemon when the other parts of SAMBA start, but it is possible to test out just the winbind portion first. To start up winbind services, enter the following command as root:

```
root# /usr/local/samba/bin/winbindd
```

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

```
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
```

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 RedHat and `/etc/init.d/samba` in Debian. 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
}
```

The 'stop' function has a corresponding entry to shut down the services and look s like this:

```
stop() {
    KIND="SMB"
    echo -n $"Shutting down $KIND services: "
    killproc smbd
    RETVAL=$?
```

SAMBA Project Documentation

```
    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

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 smbd
```



```

killproc winbindd
;;

*)
echo "Usage: /etc/init.d/samba.server { start | stop }"
;;
esac

```

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.

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`). RedHat 7.1 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

SAMBA Project Documentation

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 one
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 that 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
```

SAMBA Project Documentation

```
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/$ISA/pam_unix.so.1
dtssession auth required /usr/lib/security/$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 & try connecting through your application that you configured in the `pam.conf`.

Limitations

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 operating system, 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 been set for Windows NT users.
-

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.

How to Configure Samba 2.2 as a Primary Domain Controller

Prerequisite Reading

Before you continue reading in this chapter, please make sure that you are comfortable with configuring basic files services in `smb.conf` and how to enable and administer password encryption in Samba. These two topics are covered in the [smb.conf\(5\)](#) manpage and the [Encryption chapter](#) of this HOWTO Collection.

Background

Author's Note: This document is a combination of David Bannon's "Samba 2.2 PDC HOWTO" and "Samba NT Domain FAQ". Both documents are superseded by this one.

Versions of Samba prior to release 2.2 had marginal capabilities to act as a Windows NT 4.0 Primary Domain Controller (PDC). With Samba 2.2.0, we are proud to announce official support for Windows NT 4.0–style domain logons from Windows NT 4.0 and Windows 2000 clients. This article outlines the steps necessary for configuring Samba as a PDC. It is necessary to have a working Samba server prior to implementing the PDC functionality. If you have not followed the steps outlined in [UNIX_INSTALL.html](#), please make sure that your server is configured correctly before proceeding. Another good resource in the [smb.conf\(5\) man page](#). The following functionality should work in 2.2:

- domain logons for Windows NT 4.0/2000 clients.
- placing a Windows 9x client in user level security
- retrieving a list of users and groups from a Samba PDC to Windows 9x/NT/2000 clients
- roving (roaming) user profiles
- Windows NT 4.0–style system policies

The following pieces of functionality are not included in the 2.2 release:

- Windows NT 4 domain trusts
- SAM replication with Windows NT 4.0 Domain Controllers (i.e. a Samba PDC and a Windows NT BDC or vice versa)
- Adding users via the User Manager for Domains
- Acting as a Windows 2000 Domain Controller (i.e. Kerberos and Active Directory)

Please note that Windows 9x clients are not true members of a domain for reasons outlined in this article. Therefore the protocol for support Windows 9x–style domain logons is completely different from NT4 domain logons and has been officially supported for some time.

Implementing a Samba PDC can basically be divided into 2 broad steps.

1. Configuring the Samba PDC
2. Creating machine trust accounts and joining clients to the domain

There are other minor details such as user profiles, system policies, etc... However, these are not necessarily specific to a Samba PDC as much as they are related to Windows NT networking concepts. They will be mentioned only briefly here.

Configuring the Samba Domain Controller

The first step in creating a working Samba PDC is to understand the parameters necessary in `smb.conf`. I will not attempt to re-explain the parameters here as they are more than adequately covered in [the `smb.conf` man page](#). For convenience, the parameters have been linked with the actual `smb.conf` description.

Here is an example `smb.conf` for acting as a PDC:

```
[global]
; Basic server settings
netbios_name = POGO
workgroup = NARNIA

; we should act as the domain and local master browser
os_level = 64
preferred_master = yes
domain_master = yes
local_master = yes

; security settings (must user security = user)
security = user

; encrypted passwords are a requirement for a PDC
encrypt_passwords = yes

; support domain logons
domain_logons = yes

; where to store user profiles?
logon_path = \\%N\profiles\%u

; where is a user's home directory and where should it
; be mounted at?
logon_drive = H:
logon_home = \\homeserver\%u

; specify a generic logon script for all users
; this is a relative **DOS** path to the [netlogon] share
logon_script = logon.cmd

; necessary share for domain controller
[netlogon]
path = /usr/local/samba/lib/netlogon
read_only = yes
write_list = ntadmin

; share for storing user profiles
[profiles]
path = /export/smb/ntprofile
read_only = no
create_mask = 0600
directory_mask = 0700
```

There are a couple of points to emphasize in the above configuration.

- Encrypted passwords must be enabled. For more details on how to do this, refer to [ENCRYPTION.html](#).

- The server must support domain logons and a [netlogon] share
- The server must be the domain master browser in order for Windows client to locate the server as a DC. Please refer to the various Network Browsing documentation included with this distribution for details.

As Samba 2.2 does not offer a complete implementation of group mapping between Windows NT groups and Unix groups (this is really quite complicated to explain in a short space), you should refer to the [domain admin group](#) smb.conf parameter for information of creating "Domain Admins" style accounts.

Creating Machine Trust Accounts and Joining Clients to the Domain

A machine trust account is a Samba account that is used to authenticate a client machine (rather than a user) to the Samba server. In Windows terminology, this is known as a "Computer Account."

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 and 2000 clients use machine trust accounts, but Windows 9x clients do not. Hence, a Windows 9x 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 PDC stores each machine trust account in the Windows Registry. A Samba PDC, however, stores each machine trust account in two parts, as follows:

- A Samba account, stored in the same location as user LanMan and NT password hashes (currently `smbpasswd`). The Samba account possesses and uses only the NT password hash.
- A corresponding Unix account, typically stored in `/etc/passwd`. (Future releases will alleviate the need to create `/etc/passwd` entries.)

There are two ways to create machine trust accounts:

- Manual creation. Both the Samba and corresponding Unix account are created by hand.
- "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.

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 other '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 100 -d /dev/null -c "machine nickname" -s /bin/false
machine_name$
```

```
root# passwd -l machine_name$
```

SAMBA Project Documentation

The `/etc/passwd` entry will list the machine name with a "\$" appended, won't 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:505:501: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(8)`** 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 which 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!

"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 user script` option in `smb.conf`. This method is not required, however; corresponding Unix accounts may also be created manually.

Below is an example for a RedHat 6.2 Linux system.

```
[global]
# <...remainder of parameters...>
add user script = /usr/sbin/useradd -d /dev/null -g 100 -s /bin/false -M %u
```

Joining the Client to the Domain

The procedure for joining a client to the domain varies with the version of Windows.

- *Windows 2000*

When the user elects to join the client to a domain, Windows prompts for an account and password that is privileged to join the domain. A Samba administrative 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. The password for this account should be set to a different password than the associated `/etc/passwd` entry, for security reasons.

The session key of the Samba administrative 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.

- *Windows NT*

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 administrative account when prompted).

Common Problems and Errors

- *I cannot include a '\$' in a machine name.*

A 'machine name' in (typically) `/etc/passwd` of the machine name with a '\$' appended. FreeBSD (and other BSD systems?) won't create a user with a '\$' in their name.

The problem is only in the program used to make the entry, once made, it works perfectly. So create a user without the '\$' and use **vi** to edit the entry, adding the '\$'. Or create the whole entry with **vi** if you like, make sure you use a unique User ID !

- *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:\WINNT\> 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.

- *The system can not 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 can not log you on (C000019B), Please try again or consult your system administrator" when attempting to logon.

This occurs when the domain SID stored in `private/WORKGROUP.SID` is changed. For example, you remove the file and **smbd** automatically creates a new one. Or you are swapping back and forth

between versions 2.0.7, TNG and the HEAD branch code (not recommended). 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 machine trust account for this computer either does not exist or 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 user script` method to create accounts then this would indicate that it has not worked. Ensure the domain admin user system is working.

Alternatively 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 reported that inconsistent subnet masks between the Samba server and the NT client have caused this problem. Make sure that these are consistent for both client and server.

- *When I attempt to login to a Samba Domain from a NT4/W2K workstation, I get a message about my account being disabled.*

This problem is caused by a PAM related bug in Samba 2.2.0. This bug is fixed in 2.2.1. Other symptoms could be unaccessible shares on NT/W2K member servers in the domain or the following error in your `smbd.log`: `passdb/pampass.c:pam_account(268) PAM: UNKNOWN ERROR for User: %user%`

At first be ensure to enable the useraccounts with `smbpasswd -e %user%`, this is normally done, when you create an account.

In order to work around this problem in 2.2.0, configure the `account` control flag in `/etc/pam.d/samba` file as follows:

```
account required          pam_permit.so
```

If you want to remain backward compatibility to samba 2.0.x use `pam_permit.so`, it's also possible to use `pam_pwdb.so`. There are some bugs if you try to use `pam_unix.so`, if you need this, be ensure to use the most recent version of this file.

System Policies and Profiles

Much of the information necessary to implement System Policies and Roving User Profiles in a Samba domain is the same as that for implementing these same items in a Windows NT 4.0 domain. You should read the white paper [Implementing Profiles and Policies in Windows NT 4.0](#) available from Microsoft.

Here are some additional details:

- *What about Windows NT Policy Editor?*

SAMBA Project Documentation

To create or edit `ntconfig.pol` you must use the NT Server Policy Editor, **poledit.exe** which is included with NT Server but *not NT Workstation*. There is a Policy Editor on a NTws but it is not suitable for creating *Domain Policies*. Further, although the Windows 95 Policy Editor can be installed on an NT Workstation/Server, it will not work with NT policies because the registry key that are set by the policy templates. However, the files from the NT Server will run happily enough on an NTws. You need `poledit.exe`, `common.adm` and `winnt.adm`. It is convenient to put the two *.adm files in `c:\winnt\inf` which is where the binary will look for them unless told otherwise. Note also that that directory is '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 **servicepackname /x**, i.e. 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.

- *Can Win95 do Policies?*

Install the group policy handler for Win9x to pick up group policies. Look on the Win98 CD in `\tools\reskit\netadmin\poledit`. Install group policies on a Win9x client by double-clicking `grouppol.inf`. Log off and on again a couple of times and see if Win98 picks up group policies. Unfortunately this needs to be done on every Win9x machine that uses group policies....

If group policies don't work one reports suggests getting the updated (read: working) `grouppol.dll` for Windows 9x. The group list is grabbed from `/etc/group`.

- *How do I get 'User Manager' and 'Server Manager'?*

Since I don't need to buy an NT Server CD now, how do I get the 'User Manager for Domains', the 'Server Manager'?

Microsoft distributes a version of these tools called nexus for installation on Windows 95 systems. The tools set includes

- ◆ Server Manager
- ◆ User Manager for Domains
- ◆ Event Viewer

Click here to download the archived file <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 ftp from <ftp://ftp.microsoft.com/Softlib/MSLFILES/SRVTOOLS.EXE>

What other help can I get?

There are many sources of information available in the form of mailing lists, RFC's and documentation. The docs that come with the samba distribution contain very good explanations of general SMB topics such as browsing.

- *What are some diagnostics tools I can use to debug the domain logon process and where can I find them?*

SAMBA Project Documentation

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 what 'debug level' at which to run. See the man pages on `smbd`, `nmbd` and `smb.conf` for more information on 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. In order 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, on the first time you join the domain) to generate a 'LsaEnumTrustedDomains'. Thereafter, the workstation maintains an open connection, and therefore 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 `gdb` attach and continue.

Some useful samba commands worth investigating:

- ◆ `testparm | more`
- ◆ `smbclient -L //{netbios name of server}`

An SMB enabled version of `tcpdump` is available from <http://www.tcpdup.org/>. Ethereal, another good packet sniffer for Unix and Win32 hosts, can be downloaded from <http://www.ethereal.com>.

For tracing things on the Microsoft Windows NT, Network Monitor (aka. `netmon`) is available on the Microsoft Developer Network CD's, the Windows NT Server install CD and the SMS CD's. 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.

- *How do I install 'Network Monitor' on an NT Workstation or a Windows 9x box?*

Installing `netmon` on an NT workstation requires a couple of steps. The following are 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 version of Windows NT / `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

- ◆ Goto Start – Settings – Control Panel – Network – Services – Add
- ◆ Select the 'Network Monitor Tools and Agent' and click on 'OK'.
- ◆ Click '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 DLL's for parsing the `netmon` packet dump, and `captures\`.

In order to install the `Netmon` tools on an NT Workstation, you will first need to install the 'Network Monitor Agent' from the Workstation install CD.

- ◆ Goto Start – Settings – Control Panel – Network – Services – Add
- ◆ Select the 'Network Monitor Agent' and click on 'OK'.
- ◆ Click 'OK' on the Network Control Panel.
- ◆ Insert the Windows NT Workstation 4.0 install CD when prompted.

SAMBA Project Documentation

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.

To install Netmon on a Windows 9x box install the network monitor agent from the Windows 9x 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.

- The following is a list of helpful URLs and other links:
 - ◆ Home of Samba site <http://samba.org>. We have a mirror near you !
 - ◆ The *Development* document on the Samba mirrors might mention your problem. If so, it might mean that the developers are working on it.
 - ◆ See how Scott Merrill simulates a BDC behavior at <http://www.skippy.net/linux/smb-howto.html>.
 - ◆ Although 2.0.7 has almost had its day as a PDC, David Bannon will keep the 2.0.7 PDC pages at <http://bioserve.latrobe.edu.au/samba> going for a while yet.
 - ◆ Misc links to CIFS information <http://samba.org/cifs/>
 - ◆ NT Domains for Unix <http://mailhost.cb1.com/~lkcl/ntdom/>
 - ◆ FTP site for older SMB specs: <ftp://ftp.microsoft.com/developr/drg/CIFS/>
- *How do I get help from the mailing lists?*

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 then 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 don't post questions about Samba-TNG to the main stream Samba lists.

If you post a message to one of the lists please observe the following guide lines :

- ◆ 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 time lines are 'best guess' and nothing more.
- ◆ Always mention what version of samba you are using and what operating system its running under. You should probably 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 question clear and brief, lots of long, convoluted questions get deleted before they are completely read ! Don't post html encoded messages (if you can select colour or font size its html).
- ◆ If you run one of those nifty 'I'm on holidays' things when you are away, make sure its configured to not answer mailing lists.
- ◆ Don't cross post. Work out which is the best list to post to and see what happens, i.e. don't 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, will forward it on for you.
- ◆ You might include *partial* log files written at a debug level set to as much as 20. Please don't send the entire log but enough to give the context of the error messages.

- ◆ (Possibly) 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?
- *How do I get off the mailing lists?*

To have your name removed from a samba mailing list, go to the same place you went to to get on it. Go to <http://lists.samba.org>, click on your nearest mirror and then click on **Support** and then click on **Samba related mailing lists**. Or perhaps see [here](#)

Please don't post messages to the list asking to be removed, you will just be referred to the above address (unless that process failed in some way...)

Domain Control for Windows 9x/ME

The following section contains much of the original DOMAIN.txt file previously included with Samba. Much of the material is based on what went into the book *Special Edition, Using Samba*, by Richard Sharpe.

A domain and a workgroup are exactly the same thing 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 (NT server and other systems based on NT server support this, as does at least Samba TNG now).

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 BROWSING.txt. 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 will be the focus of this section.

When an SMB client in a domain wishes to logon it broadcast 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 very stupid) 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.

Before launching into the configuration instructions, it is worthwhile looking 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 then connects to that server, logs on (does an SMBsesssetupX) and then connects to the IPC\$ share (using an SMBtconX).

3. The client then 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 this and if it is found and can be read, is retrieved and executed by the client. After this, the client disconnects from the NetLogon share.
 5. The client then 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 the user's home share, profiles for Win9X clients MUST reside in the user home directory.
 6. The client then 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\profile. 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.
-

Configuration Instructions: Network Logons

The main difference between a PDC and a Windows 9x logon server configuration is that

- Password encryption is not required for a Windows 9x logon server.
- Windows 9x/ME clients do not possess machine trust accounts.

Therefore, a Samba PDC will also act as a Windows 9x logon server.

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 or not it is ok to configure Samba as a Domain Controller in security modes other than USER. The only security mode which will not work due to technical reasons is SHARE mode security. DOMAIN and SERVER mode security is really just a variation on SMB user level security.

Actually, this issue is also closely tied to the debate on whether or not 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 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. For this reason, it is very 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, then it is a fact that some other machine on the network (the "password server") knows more about user than the Samba host. 99% of the time, this other host is a domain controller. Now in order 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, right?)

Therefore 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.

Configuration Instructions: Setting up Roaming User Profiles

NOTE! Roaming profiles support is different for Win9X and WinNT.

SAMBA Project Documentation

Before discussing how to configure roaming profiles, it is useful to see how Win9X and WinNT clients implement these features.

Win9X 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 Win9X profiles are restricted to being in the user's home directory.

WinNT clients send a NetSAMLogon RPC request, which contains many fields, including a separate field for the location of the user's profiles. This means that support for profiles is different for Win9X and WinNT.

Windows NT Configuration

To support WinNT clients, in the [global] section of smb.conf set the following (for example):

```
logon path = \\profiles\server\profileshare\profilepath\%U\moreprofilepath
```

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 specified in the logon path browseable.

[lkcl 26aug96 – we have discovered a problem where Windows clients can maintain a connection to the [homes] share in between logins. The [homes] share must NOT therefore be used in a profile path.]

Windows 9X Configuration

To support Win9X clients, you must use the "logon home" parameter. Samba has now been fixed so that "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 Win9X 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 Win9X clients will dutifully put their clients in a subdirectory of your home directory called .profiles (thus making them hidden).

Not only that, but 'net use/home' will also work, because of a feature in Win9X. 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".

Win9X and WinNT Configuration

You can support profiles for both Win9X and WinNT clients by setting both the "logon home" and "logon path" parameters. For example:

```
logon home = \\%L%\%U\.profiles
logon path = \\%L%\profiles\%U
```

I have not checked what 'net use /home' does on NT when "logon home" is set as above.

Windows 9X 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 95 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 95 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 95, 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, if you ask me.

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 95 machine will inform you that 'The user has not logged on before' and asks you if you wish to save the user's preferences? Select 'yes'.

Once the Windows 95 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 short-cuts, 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 short-cuts from each set.

If you have made the folders / files read-only on the samba server, then you will get errors from the w95 machine on logon and logout, as it attempts to merge the local and the remote profile. Basically, if you have any errors reported by the w95 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 then next logs in, they will be told that they are 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

SAMBA Project Documentation

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.

[Exit the registry editor].

3. *WARNING* – before deleting the contents of the directory listed in the ProfilePath (this is likely to be c:\windows\profiles\username), ask them 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.

4. search for the user's .PWL password-caching file in the c:\windows directory, and delete it.
5. log off the windows 95 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.

If all else fails, increase samba's debug log levels to between 3 and 10, and / or run a packet trace program such as tcpdump or netmon.exe, and look for any error reports.

If you have access to an NT server, then first set up roaming profiles and / or netlogons on the NT server. Make a packet trace, or examine the example packet traces provided with NT server, and see what the differences are with the equivalent samba trace.

Windows NT Workstation 4.0

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.

[lkcl 10aug97 – i tried setting the path to \\samba-server\homes\profile, and discovered that this fails because a background process maintains the connection to the [homes] share which does not close down in between user logins. you have to have \\samba-server%\%L\profile, where user is the username created from the [homes] share].

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 NT 4.0 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) [lkcl 10aug97 – i found that the creation of the .PDS directory failed, and had to create these manually for each user, with a shell script. also, i presume, but have not tested, that the full profile path must be browseable just as it is for w95, due to the manner in which they attempt to create the full profile path: test existence of each path component; create path component].

In the profile directory, NT creates more folders than 95. 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.

SAMBA Project Documentation

[lkcl 10aug97 – i notice that NT Workstation tells me that it is downloading a profile from a slow link. whether this is actually the case, or whether there is some configuration issue, as yet unknown, that makes NT Workstation *_think_* that the link is a slow one is a matter to be resolved].

[lkcl 20aug97 – after samba digest correspondence, one user found, and another confirmed, that profiles cannot be loaded from a samba server unless "security = user" and "encrypt passwords = yes" (see the file ENCRYPTION.txt) or "security = server" and "password server = ip.address. of.yourNTserver" are used. Either of these options will allow the NT workstation to access the samba server using LAN manager encrypted passwords, without the user intervention normally required by NT workstation for clear-text passwords].

[lkcl 25aug97 – more comments received about NT profiles: the case of the profile *_matters_*. the file *_must_* be called NTuser.DAT or, for a mandatory profile, NTuser.MAN].

Windows NT Server

There is nothing to stop you 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.

Sharing Profiles between W95 and NT Workstation 4.0

Potentially outdated or incorrect material follows

I think this is all bogus, but have not deleted it. (Richard Sharpe)

The default logon path is \\%N%U. NT Workstation will attempt to create a directory "\\samba-server\username.PDS" if you specify the logon path as "\\samba-server\username" with the NT User Manager. Therefore, you will need to specify (for example) "\\samba-server\username\profile". NT 4.0 will attempt to create "\\samba-server\username\profile.PDS", which is more likely to succeed.

If you then want to share the same Start Menu / Desktop with W95, you will need to specify "logon path = \\samba-server\username\profile" [lkcl 10aug97 this has its drawbacks: i created a shortcut to telnet.exe, which attempts to run from the c:\winnt\system32 directory. this directory is obviously unlikely to exist on a Win95-only host].

If you have this set up correctly, you will find separate user.DAT and NTuser.DAT files in the same profile directory.

[lkcl 25aug97 – there are some issues to resolve with downloading of NT profiles, probably to do with time/date stamps. i have found that NTuser.DAT is never updated on the workstation after the first time that it is copied to the local workstation profile directory. this is in contrast to w95, where it *_does_* transfer / update profiles correctly].

DOMAIN_CONTROL.txt : Windows NT Domain Control & Samba

Possibly Outdated Material

SAMBA Project Documentation

This appendix was originally authored by John H Terpstra of the Samba Team and is included here for posterity.

NOTE : The term "Domain Controller" and those related to it refer to one specific method of authentication that can underly an SMB domain. Domain Controllers prior to Windows NT Server 3.1 were sold by various companies and based on private extensions to the LAN Manager 2.1 protocol. Windows NT introduced Microsoft-specific ways of distributing the user authentication database. See DOMAIN.txt for examples of how Samba can participate in or create SMB domains based on shared authentication database schemes other than the Windows NT SAM.

Windows NT Server can be installed as either a plain file and print server (WORKGROUP workstation or server) or as a server that participates in Domain Control (DOMAIN member, Primary Domain controller or Backup Domain controller). The same is true for OS/2 Warp Server, Digital Pathworks and other similar products, all of which can participate in Domain Control along with Windows NT.

To many people these terms can be confusing, so let's try to clear the air.

Every Windows NT system (workstation or server) has a registry database. The registry contains entries that describe the initialization information for all services (the equivalent of Unix Daemons) that run within the Windows NT environment. The registry also contains entries that tell application software where to find dynamically loadable libraries that they depend upon. In fact, the registry contains entries that describes everything that anything may need to know to interact with the rest of the system.

The registry files can be located on any Windows NT machine by opening a command prompt and typing:

```
C:\WINNT\> dir %SystemRoot%\System32\config
```

The environment variable %SystemRoot% value can be obtained by typing:

```
C:\WINNT>echo %SystemRoot%
```

The active parts of the registry that you may want to be familiar with are the files called: default, system, software, sam and security.

In a domain environment, Microsoft Windows NT domain controllers participate in replication of the SAM and SECURITY files so that all controllers within the domain have an exactly identical copy of each.

The Microsoft Windows NT system is structured within a security model that says that all applications and services must authenticate themselves before they can obtain permission from the security manager to do what they set out to do.

The Windows NT User database also resides within the registry. This part of the registry contains the user's security identifier, home directory, group memberships, desktop profile, and so on.

Every Windows NT system (workstation as well as server) will have its own registry. Windows NT Servers that participate in Domain Security control have a database that they share in common – thus they do NOT own an independent full registry database of their own, as do Workstations and plain Servers.

The User database is called the SAM (Security Access Manager) database and is used for all user authentication as well as for authentication of inter- process authentication (i.e. to ensure that the service

SAMBA Project Documentation

action a user has requested is permitted within the limits of that user's privileges).

The Samba team have produced a utility that can dump the Windows NT SAM into smbpasswd format: see ENCRYPTION.txt for information on smbpasswd and /pub/samba/pwdump on your nearest Samba mirror for the utility. This facility is useful but cannot be easily used to implement SAM replication to Samba systems.

Windows for Workgroups, Windows 95, and Windows NT Workstations and Servers can participate in a Domain security system that is controlled by Windows NT servers that have been correctly configured. Almost every domain will have ONE Primary Domain Controller (PDC). It is desirable that each domain will have at least one Backup Domain Controller (BDC).

The PDC and BDCs then participate in replication of the SAM database so that each Domain Controlling participant will have an up to date SAM component within its registry.

How to Act as a Backup Domain Controller in a Purely Samba Controlled Domain

Prerequisite Reading

Before you continue reading in this chapter, please make sure that you are comfortable with configuring a Samba PDC as described in the [Samba-PDC-HOWTO](#).

Background

What is a Domain Controller? It is a machine that is able to answer logon requests from workstations in a Windows NT Domain. Whenever a user logs into a Windows NT Workstation, the workstation connects to a Domain Controller and asks him whether the username and password the user typed in is correct. The Domain Controller replies with a lot of information about the user, for example the place where the users profile is stored, the users full name of the user. All this information is stored in the NT user database, the so-called SAM.

There are two kinds of Domain Controller in a NT 4 compatible Domain: A Primary Domain Controller (PDC) and one or more Backup Domain Controllers (BDC). The PDC contains the master copy of the SAM. Whenever the SAM has to change, for example when a user changes his password, this change has to be done on the PDC. A Backup Domain Controller is a machine that maintains a read-only copy of the SAM. This way it is able to reply to logon requests and authenticate users in case the PDC is not available. During this time no changes to the SAM are possible. Whenever changes to the SAM are done on the PDC, all BDC receive the changes from the PDC.

Since version 2.2 Samba officially supports domain logons for all current Windows Clients, including Windows 2000 and XP. This text assumes the domain to be named SAMBA. To be able to act as a PDC, some parameters in the [global]-section of the smb.conf have to be set:

```
workgroup = SAMBA
domain master = yes
domain logons = yes
```

Several other things like a [homes] and a [netlogon] share also may be set along with settings for the profile path, the users home drive and others. This will not be covered in this document.

What qualifies a Domain Controller on the network?

Every machine that is a Domain Controller for the domain SAMBA has to register the NetBIOS group name SAMBA#1c with the WINS server and/or by broadcast on the local network. The PDC also registers the unique NetBIOS name SAMBA#1b with the WINS server. The name type #1b 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.

How does a Workstation find its domain controller?

A NT workstation in the domain SAMBA that wants a local user to be authenticated has to find the domain controller for SAMBA. It does this by doing a NetBIOS name query for the group name SAMBA#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 (TODO: How is the DC chosen) domain controller authenticate each other. After that the workstation sends the user's credentials (his name and password) to the domain controller, asking for approval.

When is the PDC needed?

Whenever a user wants to change his password, this has to be done on the PDC. To find the PDC, the workstation does a NetBIOS name query for SAMBA#1b, assuming this machine maintains the master copy of the SAM. The workstation contacts the PDC, both mutually authenticate and the password change is done.

Can Samba be a Backup Domain Controller?

With version 2.2, no. The native NT SAM replication protocols have not yet been fully implemented. The Samba Team is working on understanding and implementing the protocols, but this work has not been finished for version 2.2.

Can I get the benefits of a BDC with Samba? Yes. 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.

How do I set up a Samba BDC?

Several things have to be done:

- The domain SID has to be the same on the PDC and the BDC. This used to be stored in the file `private/MACHINE.SID`. This file is not created anymore since Samba 2.2.5 or even earlier. Nowadays the domain SID is stored in the file `private/secrets.tdb`. Simply copying the `secrets.tdb` from the PDC to the BDC does not work, as the BDC would generate a new SID for itself and override the domain SID with this new BDC SID.

To retrieve the domain SID from the PDC or an existing BDC and store it in the `secrets.tdb`, execute `'net rpc getsid'` on the BDC.
- 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, or the PDC is set up as a NIS master server and the BDC as a 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.
- The Samba password database in the file `private/smbpasswd` has to be replicated from the PDC to the BDC. This is a bit tricky, see the next section.
- Any 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 together with the `smbpasswd` synchronization.

Finally, the BDC has to be found by the workstations. This can be done by setting

```
workgroup = samba
domain master = no
domain logons = yes
```

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.

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.

Storing Samba's User/Machine Account information in an LDAP Directory

Purpose

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.

- OpenLDAP – <http://www.openldap.org/>
- iPlanet Directory Server – <http://iplanet.netscape.com/directory>

Note that O'Reilly Publishing is working on a guide to LDAP for System Administrators which has a planned release date of early summer, 2002.

Two additional Samba resources which may prove to be helpful are

- The Samba-PDC-LDAP-HOWTO maintained by Ignacio Coupeau.
- The NT migration scripts from IDEALX that are geared to manage users and group in such a Samba-LDAP Domain Controller configuration.

Introduction

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 very large numbers of users (counted in the thousands).

- The first 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 is used in databases.
- The second problem is that administrators who desired 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.
- And finally, the amount of information which is stored in an `smbpasswd` entry leaves no room for additional attributes such as a home directory, password expiration time, or even a Relative Identified (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 CVS trees). In Samba 2.2.3, enabling support for a `samdb` backend (e.g. `--with-ldapsam` or `--with-tdbsam`) requires compile time support.

When compiling Samba to include the `--with-ldapsam` autoconf option, `smbd` (and associated tools) will store and lookup user accounts in an LDAP directory. In reality, this is very easy to understand. If you are comfortable with using an `smbpasswd` file, simply replace "smbpasswd" with "LDAP directory" in all the

documentation.

There are a few points to stress about what the `--with-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 2000 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 Software (<http://www.padl.com/>). However, the details of configuring these packages are beyond the scope of this document.

Supported LDAP Servers

The LDAP `samdb` code in 2.2.3 has been developed and tested using the OpenLDAP 2.0 server and client libraries. The same code should be able to work with Netscape's Directory Server and client SDK. However, due to lack of testing so far, there are bound to be compile errors and bugs. These should not be hard to fix. If you are so inclined, please be sure to forward all patches to samba-patches@samba.org and jerry@samba.org.

Schema and Relationship to the RFC 2307 `posixAccount`

Samba 2.2.3 includes the necessary schema file for OpenLDAP 2.0 in `examples/LDAP/samba.schema`. (Note that this schema file has been modified since the experimental support initially included in 2.2.2). The `sambaAccount` objectclass is given here:

```
objectclass ( 1.3.1.5.1.4.1.7165.2.2.2 NAME 'sambaAccount' SUP top STRUCTURAL
  DESC 'Samba Account'
  MUST ( uid $ rid )
  MAY ( cn $ lmPassword $ ntPassword $ pwdLastSet $ logonTime $
    logoffTime $ kickoffTime $ pwdCanChange $ pwdMustChange $ acctFlags $
    displayName $ smbHome $ homeDrive $ scriptPath $ profilePath $
    description $ userWorkstations $ primaryGroupID $ domain ) )
```

The `samba.schema` file has been formatted for OpenLDAP 2.0. The OID's are owned by the Samba Team and as such is legal to be openly published. If you translate the schema to be used with Netscape DS, please submit the modified schema file as a patch to jerry@samba.org

Just as the `smbpasswd` file is meant to store information which supplements a user's `/etc/passwd` entry, so is the `sambaAccount` object meant to supplement the UNIX user account information. A `sambaAccount` is a `STRUCTURAL` objectclass so it can be stored individually in the directory. However, there are several fields (e.g. `uid`) which 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 `sambaAccount` 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.

Configuring Samba with LDAP

OpenLDAP configuration

To include support for the sambaAccount object in an OpenLDAP directory server, first copy the samba.schema file to slapd's configuration directory.

```
root# cp samba.schema /etc/openldap/schema/
```

Next, include the samba.schema file in slapd.conf. The sambaAccount object contains two attributes which depend upon 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 sambaAccount
include          /etc/openldap/schema/cosine.schema
include          /etc/openldap/schema/inetorgperson.schema
include          /etc/openldap/schema/samba.schema

## uncomment this line if you want to support the RFC2307 (NIS) schema
## include       /etc/openldap/schema/nis.schema

....
```

It is recommended that you maintain some indices on some of the most useful attributes, like in the following example, to speed up searches made on sambaAccount objectclasses (and possibly posixAccount and posixGroup as well).

```
# Indices to maintain
## required by OpenLDAP 2.0
index objectclass eq

## support pb_getsampwnam()
index uid          pres,eq
## support pdb_getsambapwrid()
index rid          eq

## uncomment these if you are storing posixAccount and
## posixGroup entries in the directory as well
##index uidNumber eq
##index gidNumber eq
##index cn         eq
##index memberUid eq
```

Configuring Samba

The following parameters are available in smb.conf only with `--with-ldapsam` was included with compiling Samba.

- [ldap ssl](#)
- [ldap server](#)
- [ldap admin dn](#)
- [ldap suffix](#)
- [ldap filter](#)
- [ldap port](#)

These are described in the [smb.conf\(5\)](#) man page and so will not be repeated here. However, a sample smb.conf file for use with an LDAP directory could appear as

```
## /usr/local/samba/lib/smb.conf
[global]
    security = user
    encrypt passwords = yes

    netbios name = TASHTEGO
    workgroup = NARNIA

    # 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
    # changes, this password will need to be reset.
    ldap admin dn = "cn=Samba Manager,ou=people,dc=samba,dc=org"

    # specify the LDAP server's hostname (defaults to localhost)
    ldap server = ahab.samba.org

    # Define the SSL option when connecting to the directory
    # ('off', 'start tls', or 'on' (default))
    ldap ssl = start tls

    # define the port to use in the LDAP session (defaults to 636 when
    # "ldap ssl = on")
    ldap port = 389

    # specify the base DN to use when searching the directory
    ldap suffix = "ou=people,dc=samba,dc=org"

    # generally the default ldap search filter is ok
    # ldap filter = "(&(uid=%u)(objectclass=sambaAccount))"
```

Accounts and Groups management

As users accounts are managed thru the sambaAccount objectclass, you should modify you existing administration tools to deal with sambaAccount attributes.

Machines accounts are managed with the sambaAccount objectclass, just like users accounts. However, it's up to you to stored thoses accounts in a different tree of you LDAP namespace: you should use "ou=Groups,dc=plainjoe,dc=org" to store groups and "ou=People,dc=plainjoe,dc=org" to store users. Just configure your NSS and PAM accordingly (usually, in the /etc/ldap.conf configuration file).

In Samba release 2.2.3, the group management system is based on posix groups. This meand that Samba make usage of the posixGroup objectclass. For now, there is no NT-like group system management (global and

local groups).

Security and sambaAccount

There are two important points to remember when discussing the security of sambaAccount 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 clear text equivalents and can be used to impersonate the user without deriving the original clear text strings. For more information on the details of LM/NT password hashes, refer to the [ENCRYPTION chapter](#) of the Samba-HOWTO-Collection.

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 2.0 server, it is possible to use the use the StartTLS 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=plainjoe,dc=org" write
    by * none
```

LDAP specials attributes for sambaAccounts

The sambaAccount objectclass is composed of the following attributes:

- **lmPassword**: the LANMAN password 16-byte hash stored as a character representation of a hexadecimal string.
- **ntPassword**: the NT password hash 16-byte stored as a character representation of a hexadecimal string.
- **pwdLastSet**: The integer time in seconds since 1970 when the lmPassword and ntPassword attributes were last set.
- **acctFlags**: string of 11 characters surrounded by square brackets [] representing account flags such as U (user), W(workstation), X(no password expiration), and D(disabled).
- **logonTime**: Integer value currently unused
- **logoffTime**: Integer value currently unused
- **kickoffTime**: Integer value currently unused
- **pwdCanChange**: Integer value currently unused
- **pwdMustChange**: Integer value currently unused
- **homeDrive**: specifies the drive letter to which to map the UNC path specified by homeDirectory. The drive letter must be specified in the form "X:" where X is the letter of the drive to map. Refer to

SAMBA Project Documentation

the "logon drive" parameter in the smb.conf(5) man page for more information.

- `scriptPath`: The `scriptPath` 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 "logon script" parameter in the smb.conf(5) man page for more information.
- `profilePath`: 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 "logon path" parameter in the smb.conf(5) man page for more information.
- `smbHome`: The `homeDirectory` property specifies the path of the home directory for the user. The string can be null. If `homeDrive` is set and specifies a drive letter, `homeDirectory` should be a UNC path. The path must be a network UNC path of the form `\\server\share\directory`. This value can be a null string. Refer to the "logon home" parameter in the smb.conf(5) man page for more information.
- `userWorkstation`: character string value currently unused.
- `rid`: the integer representation of the user's relative identifier (RID).
- `primaryGroupID`: the relative identifier (RID) of the primary group of the user.

The majority of these parameters are only used when Samba is acting as a PDC of a domain (refer to the [Samba-PDC-HOWTO](#) for details on how to configure Samba as a Primary Domain Controller). The following four attributes are only stored with the `sambaAccount` entry if the values are non-default values:

- `smbHome`
- `scriptPath`
- `logonPath`
- `homeDrive`

These attributes are only stored with the `sambaAccount` entry if the values are non-default values. For example, assume TASHTEGO 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 `\\TASHTEGO\becky`. If the `smbHome` 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`).

Example LDIF Entries for a sambaAccount

The following is a working LDIF with the inclusion of the `posixAccount` objectclass:

```
dn: uid=guest2, ou=people,dc=plainjoe,dc=org
ntPassword: 878D8014606CDA29677A44EFA1353FC7
pwdMustChange: 2147483647
primaryGroupID: 1201
lmPassword: 552902031BEDE9EFAAD3B435B51404EE
pwdLastSet: 1010179124
logonTime: 0
objectClass: sambaAccount
uid: guest2
kickoffTime: 2147483647
acctFlags: [UX      ]
logoffTime: 2147483647
rid: 19006
pwdCanChange: 0
```

The following is an LDIF entry for using both the `sambaAccount` and `posixAccount` objectclasses:

SAMBA Project Documentation

```
dn: uid=gcarter, ou=people,dc=plainjoe,dc=org
logonTime: 0
displayName: Gerald Carter
lmPassword: 552902031BEDE9EFAAD3B435B51404EE
primaryGroupID: 1201
objectClass: posixAccount
objectClass: sambaAccount
acctFlags: [UX      ]
userPassword: {crypt}BpM2ej8Rkzogo
uid: gcarter
uidNumber: 9000
cn: Gerald Carter
loginShell: /bin/bash
logoffTime: 2147483647
gidNumber: 100
kickoffTime: 2147483647
pwdLastSet: 1010179230
rid: 19000
homeDirectory: /home/tashtego/gcarter
pwdCanChange: 0
pwdMustChange: 2147483647
ntPassword: 878D8014606CDA29677A44EFA1353FC7
```

Comments

Please mail all comments regarding this HOWTO to jerry@samba.org. This documents was last updated to reflect the Samba 2.2.3 release.

Improved browsing in samba

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 which 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.

Browsing will NOT work if name resolution from NetBIOS names to IP addresses does not function correctly. 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 can NOT be provided by any other means of name resolution.

Browsing support in samba

Samba now fully supports browsing. The browsing is supported by nmbd and is also controlled by options in the smb.conf file (see smb.conf(5)).

Samba can act as a local browse master for a workgroup and the ability for samba to support domain logons and scripts is now available. See DOMAIN.txt for more information on domain logons.

Samba can also act as a domain master browser for a workgroup. This means that it will collate lists from local browse masters 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.

Note that you should 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 that nmbd can be configured as a WINS server, but it is not necessary to specifically use samba as your WINS server. NTAS can be configured as your WINS server. In a mixed NT server and samba environment on a Wide Area Network, it is recommended that you use the NT server's WINS server capabilities. In a samba-only environment, it is recommended that you use one and only one nmbd as your 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.

Problem resolution

If something doesn't work then hopefully the log.nmb file will help you track down the problem. Try a debug 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.

SAMBA Project Documentation

Note that if it doesn't work for you, then you should still be able to type the server name as `\\SERVER` in filemanager then hit enter and filemanager should display the list of available shares.

Some people find browsing fails because they don't 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.

Also, a lot of people are getting bitten by the problem of too many parameters on the command line of `nmbd` in `inetd.conf`. This trick is to not use spaces between the option and the parameter (eg: `-d2` instead of `-d 2`), and to not use the `-B` and `-N` options. New versions of `nmbd` are now far more likely to correctly find your broadcast and network address, so in most cases these aren't needed.

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`)

Browsing across subnets

With the release of Samba 1.9.17(alpha1 and above) Samba has been updated to enable it to support the replication of browse lists across subnet boundaries. New code and options have been added to achieve this. This section describes how to set this feature up in different settings.

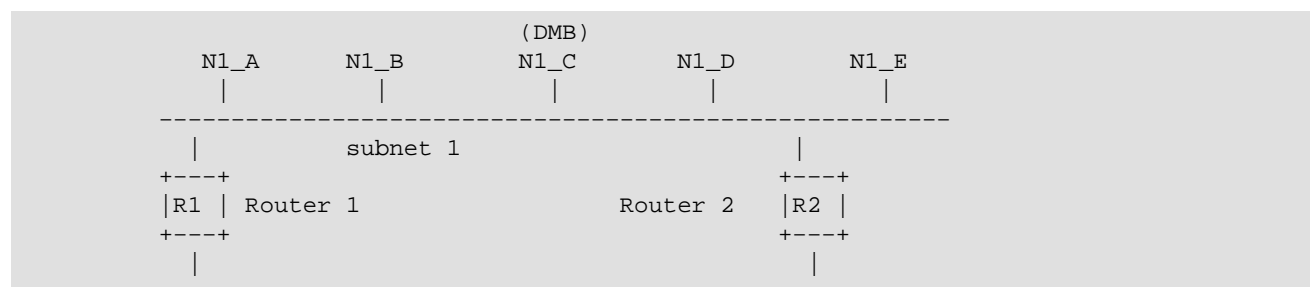
To see browse lists that span TCP/IP subnets (ie. networks separated by routers that don't pass broadcast traffic) you must set up at least one WINS server. The WINS server acts as a DNS for NetBIOS names, allowing NetBIOS name to IP address translation to be done by doing a direct query of the WINS server. This is done via a directed UDP packet on port 137 to the WINS server machine. The reason for a WINS server is that by default, all NetBIOS name to IP address translation is done by 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 Win95 and WinNT, this is in the TCP/IP Properties, under Network settings) for Samba this is in the `smb.conf` file.

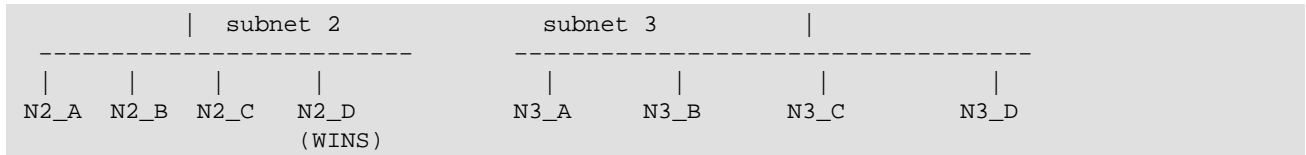
How does cross subnet browsing work ?

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. However, with the 1.9.17 release, Samba is capable of cross subnet browsing when configured correctly.

Consider a network set up as follows :



SAMBA Project Documentation



Consisting of 3 subnets (1, 2, 3) connected by two routers (R1, R2) – these do not pass broadcasts. Subnet 1 has 5 machines on it, subnet 2 has 4 machines, subnet 3 has 4 machines. Assume for the moment that all these machines are configured to be in the same workgroup (for simplicities sake). Machine N1_C on subnet 1 is configured as Domain Master Browser (ie. 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 all 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 look as follows (these are the machines you would see in your network neighborhood if you looked in it on a particular network right now).

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

Note that at this point all the subnets are separate, 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 to synchronize its browse list with. 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 booted.

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

SAMBA Project Documentation

synchronization request to the sender of that packet. After both synchronizations are done the browse lists look like :

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 a (*) 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 vice-versa the browse lists look like.

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(*)

Servers with a (*) 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, 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 – and as a steady state (if no machines are removed or shut off) the browse lists will look like :

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 a (*) after them are non-authoritative names.

Synchronizations between the domain master browser and local master browsers will continue to occur, but this should be a steady state situation.

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 neighbourhood lists.
2. Attempts to connect to these inaccessible computers will fail, but the names will not be removed from the network neighbourhood 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, by using subnet-isolated broadcast NetBIOS name resolution. The effects are similar to that of losing access to a DNS server.

Setting up a WINS server

Either a Samba machine or a Windows NT Server machine may be set up as a WINS server. To set a Samba machine to be a WINS server you must add the following option to the smb.conf file on the selected machine : in the [globals] section add the line

wins support = yes

Versions of Samba previous 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 1.9.17 or above, or at the very least set the parameter to 'no' on all these machines.

Machines with "**wins support = yes**" will keep a list of all NetBIOS names registered with them, acting as a DNS for NetBIOS names.

You should set up only ONE wins server. Do NOT set the "**wins support = yes**" option on more than one Samba server.

To set up a Windows NT Server as a WINS server you need to set up the WINS service – see your NT documentation for details. Note that Windows NT WINS Servers can replicate to each other, allowing more than one to be set up in a complex subnet environment. As Microsoft refuse to document these replication protocols Samba cannot currently participate in these replications. It is possible in the future that a Samba->Samba WINS replication protocol may be defined, in which case more than one Samba machine could be set up as a WINS server but 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 95 or Windows NT. 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.

Note that 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 95, Samba and Windows NT 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.

Setting up Browsing in a WORKGROUP

To set up cross subnet browsing on a network containing machines in up to be 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 any other subnet. It is the presence of a domain master browser that makes cross subnet browsing possible for a workgroup.

In an 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 also 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 :

```
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 you require.

Next, you should ensure that each of the subnets contains a machine that can act as a local master browser for the workgroup. Any NT machine should be able to do this, as will Windows 95 machines (although these tend to get rebooted more often, so it's 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 :

```
domain master = no
local master = yes
preferred master = yes
os level = 65
```

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 that it should win any browser elections.

If you have an NT machine on the subnet that you wish to be the local master browser then you can disable Samba from becoming a local master browser by setting the following options in the [global] section of the smb.conf file :

```
domain master = no
local master = no
preferred master = no
os level = 0
```

Setting up Browsing in a DOMAIN

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 name is also the Domain master browser for that name, and many things will 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 :

```
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 see the section "FORCING SAMBA TO BE THE MASTER" below.

If you have Windows NT machines that are members of the domain on all subnets, and you are sure they will always be running then you can disable Samba from taking part in browser elections and ever becoming a local master browser by setting following options in the [global] section of the smb.conf file :

```
domain master = no local master = no preferred master = no os level = 0
```

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 which determine what precedence (bias) a host should have in the election. By default Samba uses a very low precedence and thus loses elections to just about anyone else.

If you want Samba to win elections then just set the "os level" global option in smb.conf to a higher number. It defaults to 0. Using 34 would make it win all elections over every other system (except other samba systems!)

A "os level" of 2 would make it beat WfWg and Win95, but not NTAS. A NTAS domain controller uses level 32.

The maximum os level is 255

If you want samba to force an election on startup, then 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 95 or NT 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 5 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.

Making samba the domain master

The domain master is responsible for collating the browse lists of multiple subnets so that 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.

Note that you should NOT set Samba to be the domain master for a workgroup that has the same name as an NT Domain.

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 synchronise browse lists.

If you want samba to be the domain master then I suggest you 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.

Note that all your 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. your local master browsers will be unable to find a domain master browser, as it will only be looking 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. your 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, your 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.
-

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 0's broadcast and you will probably find that browsing and name lookups won't work.

Multiple interfaces

Samba now supports machines with multiple network interfaces. If you have multiple interfaces then you will need to use the "interfaces" option in smb.conf to configure them. See smb.conf(5) for details.

Samba performance issues

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 a NT or WfWg 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 very much depends on your system.

Several people have done comparisons between Samba and Novell, NFS or WinNT. In some cases Samba performed the best, in others the worst. I suspect the biggest factor is not Samba vs 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.

Oplocks

Overview

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.

With the release of Samba 1.9.18 we now correctly support opportunistic locks. This is turned on by default, and can be turned off on a share-by-share basis by setting the parameter :

oplocks = False

We recommend that you leave oplocks on however, as current benchmark tests with NetBench seem to give approximately a 30% improvement in speed with them on. This is on average however, and the actual improvement seen can be orders of magnitude greater, depending on what the client redirector is doing.

Previous to Samba 1.9.18 there was a 'fake oplocks' option. This option has been left in the code for backwards compatibility reasons but it's use is now deprecated. A short summary of what the old code did follows.

Level2 Oplocks

With Samba 2.0.5 a new capability – level2 (read only) oplocks is supported (although the option is off by default – see the smb.conf man page for details). Turning on level2 oplocks (on a share-by-share basis) by setting the parameter :

level2 oplocks = true

should speed concurrent access to files that are not commonly written to, such as application serving shares (ie. shares that contain common .EXE files – such as a Microsoft Office share) as it allows clients to read-ahead cache copies of these files.

Old 'fake oplocks' option – deprecated

Samba can also fake oplocks, by granting a oplock whenever a client asks for one. This is controlled using the smb.conf option "fake oplocks". If you set "fake oplocks = yes" then you are telling the client that it may aggressively cache the file data for all opens.

Enabling 'fake oplocks' on all read-only shares or shares that you know will only be accessed from one client at a time 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.

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 right 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.

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 `SMBreadbraw`) 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 `SMBreadbraw`, 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.

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 request 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 honours 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.

Locking

By default Samba does not implement strict locking on each read/write call (although it did in previous versions). If you enable strict locking (using "strict locking = yes") then you may find that you suffer a severe performance hit on some systems.

The performance hit will probably be greater on NFS mounted filesystems, but could be quite high even on local disks.

Share modes

Some people find that opening files is very slow. This is often because of the "share modes" code needed to fully implement the dos share modes stuff. You can disable this code using "share modes = no". This will gain you a lot in opening and closing files but will mean that (in some cases) the system won't force a second user of a file to open the file read-only if the first has it open read-write. For many applications that do their own locking this doesn't matter, but for some it may. Most Windows applications depend heavily on "share modes" working correctly and it is recommended that the Samba share mode support be left at the default of "on".

The share mode code in Samba has been re-written in the 1.9.17 release following tests with the Ziff-Davis NetBench PC Benchmarking tool. It is now believed that Samba 1.9.17 implements share modes similarly to Windows NT.

NOTE: In the most recent versions of Samba there is an option to use shared memory via mmap() to implement the share modes. This makes things much faster. See the Makefile for how to enable this.

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 very expensive.

Wide lines

The "wide links" option is now enabled by default, but if you disable it (for better security) then you may suffer a performance hit in resolving filenames. The performance loss is lessened if you have "getwd cache = yes", which is now the default.

Read raw

The "read raw" operation is designed to be an optimised, 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 don't 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 affect your performance. Only testing can really tell.

Write raw

The "write raw" operation is designed to be an optimised, 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.

Read prediction

Samba can do read prediction on some of the SMB commands. Read prediction means that Samba reads some extra data on the last file it read while waiting for the next SMB command to arrive. It can then respond more quickly when the next read request arrives.

This is disabled by default. You can enable it by using "read prediction = yes".

Note that read prediction is only used on files that were opened read only.

Read prediction should particularly help for those silly clients (such as "Write" under NT) which do lots of very small reads on a file.

Samba will not read ahead more data than the amount specified in the "read size" option. It always reads ahead on 1k block boundaries.

Memory mapping

Samba supports reading files via memory mapping them. On some machines this can give a large boost to performance, on others it makes not difference at all, and on some it may reduce performance.

To enable you you have to recompile Samba with the `-DUSE_MMAP` option on the `FLAGS` line of the Makefile.

Note that memory mapping is only used on files opened read only, and is not used by the "read raw" operation. Thus you may find memory mapping is more effective if you disable "read raw" using "read raw = no".

Slow Clients

One person has reported that setting the protocol to `COREPLUS` rather than `LANMAN2` gave a dramatic speed improvement (from 10k/s to 150k/s).

I suspect that his PC's (386sx16 based) were asking for more data than they could chew. I suspect a similar speed could be had by setting "read raw = no" and "max xmit = 2048", instead of changing the protocol. Lowering the "read size" might also help.

Slow Logins

Slow logins are almost always due to the password checking time. Using the lowest practical "password level" will improve things a lot. You could also enable the "UFC crypt" option in the Makefile.

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.

See your client docs for details. In particular, I have heard rumours that the WfWg options `TCPWINDOWSIZE` and `TCPSEGMENTSIZ` can have a large impact on performance.

Also note that some people have found that setting `DefaultRcvWindow` in the `[MSTCP]` section of the `SYSTEM.INI` file under WfWg to 3072 gives a big improvement. I don't know why.

My own experience with `DefaultRcvWindow` is that I get much better performance with a large value (16384 or larger). Other people have reported that anything over 3072 slows things down enourmously. One person even reported a speed drop of a factor of 30 when he went from 3072 to 8192. I don't know why.

It probably depends a lot on your hardware, and the type of unix box you have at the other end of the link.

Paul Cochrane has done some testing on client side tuning and come to the following conclusions:

Install the `W2setup.exe` file from www.microsoft.com. This is an update for the winsock stack and utilities which improve performance.

Configure the win95 TCPIP registry settings to give better perfomance. I use a program called `MTUSPEED.exe` which I got off the net. There are various other utilities of this type freely available. The setting which give the best performance for me are:

1. MaxMTU Remove
2. RWIN Remove

3. MTUAutoDiscover Disable
4. MTUBlackHoleDetect Disable
5. Time To Live Enabled
6. Time To Live – HOPS 32
7. NDI Cache Size 0

I tried virtually all of the items mentioned in the document and the only one which made a difference to me was the socket options. It turned out I was better off without any!!!!

In terms of overall speed of transfer, between various win95 clients and a DX2-66 20MB server with a crappy NE2000 compatible and old IDE drive (Kernel 2.0.30). The transfer rate was reasonable for 10 baseT.

FIXME The figures are: Put Get P166 client 3Com card: 420-440kB/s 500-520kB/s P100 client 3Com card: 390-410kB/s 490-510kB/s DX4-75 client NE2000: 370-380kB/s 330-350kB/s

I based these test on transfer two files a 4.5MB text file and a 15MB textfile. The results arn't bad considering the hardware Samba is running on. It's a crap machine!!!!

The updates mentioned in 1 and 2 brought up the transfer rates from just over 100kB/s in some clients.

A new client is a P333 connected via a 100MB/s card and hub. The transfer rates from this were good: 450-500kB/s on put and 600+kB/s on get.

Looking at standard FTP throughput, Samba is a bit slower (100kB/s upwards). I suppose there is more going on in the samba protocol, but if it could get up to the rate of FTP the performance would be quite staggering.

My Results

Some people want to see real numbers in a document like this, so here they are. I have a 486sx33 client running WfWg 3.11 with the 3.11b tcp/ip stack. It has a slow IDE drive and 20Mb of ram. It has a SMC Elite-16 ISA bus ethernet card. The only WfWg tuning I've done is to set DefaultRcvWindow in the [MSTCP] section of system.ini to 16384. My server is a 486dx3-66 running Linux. It also has 20Mb of ram and a SMC Elite-16 card. You can see my server config in the examples/tridge/ subdirectory of the distribution.

I get 490k/s on reading a 8Mb file with copy. I get 441k/s writing the same file to the samba server.

Of course, there's a lot more to benchmarks than 2 raw throughput figures, but it gives you a ballpark figure.

I've also tested Win95 and WinNT, and found WinNT gave me the best speed as a samba client. The fastest client of all (for me) is smbclient running on another linux box. Maybe I'll add those results here someday ...

Samba and other CIFS clients

This chapter contains client-specific information.

Macintosh clients?

Yes. Thursby now have a CIFS Client / Server called DAVE – see

They test it against Windows 95, Windows NT and samba for compatibility issues. At the time of writing, DAVE was at version 1.0.1. The 1.0.0 to 1.0.1 update is available as a free download from the Thursby web site (the speed of finder copies has been greatly enhanced, and there are bug-fixes included).

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, and CAP. 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

OS2 Client

How can I configure OS/2 Warp Connect or OS/2 Warp 4 as a client for Samba?

A more complete answer to this question can be found on <http://carol.wins.uva.nl/~leeuw/samba/warp.html>.

Basically, you need three components:

- The File and Print Client ('IBM Peer')
- TCP/IP ('Internet support')
- The "NetBIOS over TCP/IP" driver ('TCPBEUI')

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(s) 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 webpage mentioned above.

How can I configure OS/2 Warp 3 (not Connect), OS/2 1.2, 1.3 or 2.x for Samba?

You can use the free Microsoft LAN Manager 2.2c Client for OS/2 from <ftp://ftp.microsoft.com/BusSys/Clients/LANMAN.OS2/>. See <http://carol.wins.uva.nl/~leeuw/lanman.html> for more information on how to install and use this client. 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, don't 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.

Are there any other issues when OS/2 (any version) is used as a client?

When you do a NET VIEW or use the "File and Print Client Resource Browser", no Samba servers show up. This can be fixed by a patch from <http://carol.wins.uva.nl/~leeuw/samba/fix.html>. The patch will be included in a later version of Samba. It also fixes a couple of other problems, such as preserving long filenames when objects are dragged from the Workplace Shell to the Samba server.

How do I get printer driver download working for OS/2 clients?

First, create a share called [PRINTDRV] that is world-readable. Copy your OS/2 driver files there. Note that 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`. Then, 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.

Windows for Workgroups

Use 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, VNB.T.386, WSTCP.386, TRACERT.EXE, NETSTAT.EXE, and NBTSTAT.EXE.

Delete .pwl files after password change

WfWg does a lousy job with passwords. I find that if I change my password 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 don't do this you may find that WfWg remembers and uses the old password, even if you told it a new one.

Often WfWg will totally ignore a password you give it in a dialog box.

Configure WfW 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. ie disable Password Caching etc for use with **security = user**

Case handling of passwords

Windows for Workgroups uppercases the password before sending it to the server. Unix passwords can be case-sensitive though. Check the [smb.conf\(5\)](#) information on **password level** to specify what characters samba should try to uppercase when checking.

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 affect 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.

1. Kernel Update: KRNLUPD.EXE
2. Ping Fix: PINGUPD.EXE
3. RPC Update: RPCRTUPD.EXE
4. TCP/IP Update: VIPUPD.EXE
5. 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 also notice a significant speedup when accessing network neighborhood services.

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 likely occur if it is not.

In order to server 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, etc...). See the [smb.conf\(5\)](#) 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.

The following is a 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 Win2k SP2 client copies the security descriptor for the profile which contains the Samba server's SID, and not the domain SID. The client compares the SID for SAMBA\user and realizes it is different that 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 Win2k 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.

HOWTO Access Samba source code via CVS

Introduction

Samba is developed in an open environment. Developers use CVS (Concurrent Versioning System) 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 document is a modified version of the instructions found at <http://samba.org/samba/cvs.html>

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 and jitterbug. There are two main ways of accessing the CVS server on this host.

Access via CVSweb

You can access the source code via your favourite 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>

Access via cvs

You can also access the source code via a normal cvs client. This gives you much more control over 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 which provide a graphical interface to the sometimes mundane CVS commands. Links to these clients are also available from <http://www.cyclic.com>.

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

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
```

When it asks you for a password type **cvs**.

3. Run the command

```
cvs -d :pserver:cvs@samba.org:/cvsroot co samba
```

SAMBA Project Documentation

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 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 2.2 release code. This could be done by using the following command.

```
cvs -d :pserver:cvs@samba.org:/cvsroot co -r SAMBA_2_2 samba
```

4. Whenever you want to merge in the latest code changes use the following command from within the samba directory:

```
cvs update -d -P
```

Reporting Bugs

Introduction

The email address for bug reports is samba@samba.org

Please take the time to read this file before you submit a bug report. Also, please see if it has changed between releases, as we may be changing the bug reporting mechanism at some time.

Please also 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 about it than we can possibly answer, so you have a much higher chance of an answer 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 then it is better to send it to the Samba mailing list, as there are (at last count) 5000 other users on that list that may be able to help you.

You may also like to look through the recent mailing list archives, which are conveniently accessible on the Samba web pages at <http://samba.org/samba/>

General info

Before submitting a bug report check your config for silly errors. Look in your log files for obvious messages that tell you that you've misconfigured something and run `testparm` to test your config file for correct syntax.

Have you run through the [diagnosis](#)? This is very 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.

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 very useful. Depending on the problem a 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 `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 use:

```
log level = 10
log file = /usr/local/samba/lib/log.%m
include = /usr/local/samba/lib/smb.conf.%m
```

then 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 etc on just one machine.

The `smb.conf` entry **log level =** is synonymous with the entry **debuglevel =** that has been used in older versions of Samba and is being retained for backwards compatibility of `smb.conf` files.

As the **log level =** value is increased you will record a significantly increasing 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 VERY large volume of log data.

Internal errors

If you get a "INTERNAL ERROR" message 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` then it will probably be accompanied by a message which details the last SMB message received by `smbd`. This info is often very 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 don't have `gdb` then try "dbx". Then within the debugger use the command "where" to give a stack trace of where the problem occurred. Include this in your mail.

If you know any assembly language then 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 don't know assembly then including this info in the bug report can be useful.

Attaching to a running process

Unfortunately some unices (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.

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.

Group mapping HOWTO

Starting with Samba 3.0 alpha 2, a new group mapping function is available. The current method (likely to change) to manage the groups is a new command called **smbgroupedit**.

The first immediate reason to use the group mapping on a PDC, is that the **domain admin group** of `smb.conf` is now gone. This parameter was used to give the listed users local admin rights on their workstations. It was some magic stuff that simply worked but didn't scale very well for complex setups.

Let me explain how it works on NT/W2K, to have this magic fade away. When installing NT/W2K on a computer, the installer program creates some users and groups. Notably the 'Administrators' group, and gives to that group some privileges like the ability to change the date and time or to kill any process (or close too) running on the local machine. The 'Administrator' user is a member of the 'Administrators' group, and thus 'inherit' the 'Administrators' group privileges. If a 'joe' user is created and become a member of the 'Administrator' group, 'joe' has exactly the same rights as 'Administrator'.

When a NT/W2K machine is joined to a domain, during that phase, the "Domain Administrators" group of the PDC is added to the 'Administrators' group of the workstation. Every members of the 'Domain Administrators' group 'inherit' the rights of the 'Administrators' group when logging on the workstation.

You are now wondering how to make some of your samba PDC users members of the 'Domain Administrators' ? That's really easy.

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, your entry in `/etc/group` will look like:

```
domadm:x:502:joe, john, mary
```

3. Map this `domadm` group to the **domain admins** group by running the command:

```
smbgroupedit -c "Domain Admins" -u domadm
```

You're set, joe, john and mary are domain administrators !

Like the Domain Admins group, you can map any arbitrary Unix group to any NT group. You can also make any Unix group a domain group. For example, on a domain member machine (an NT/W2K or a samba server running winbind), you would like to give access to a certain directory to some users who are member of a group on your samba PDC. Flag that group as a domain group by running:

```
smbgroupedit -a unixgroup -td
```

You can list the various groups in the mapping database like this

```
smbgroupedit -v
```

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.

HPUX

HP's implementation of supplementary groups is, er, non-standard (for hysterical reasons). There are two group files, `/etc/group` and `/etc/login.group`; 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/login.group` (hard link doesn't work for reasons too stupid to go into here). `initgroups()` will complain if one of the groups you're in in `/etc/login.group` 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 that the programs that are failing to `initgroups()` be 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)`.

SCO Unix

If you run an old version of SCO Unix then 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`).

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, some things still won't 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 this 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$:
        clr1   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$:
        clr1   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`