



Red Hat Enterprise Linux 9

Configuring and using network file services

A guide to configuring and using network file services in Red Hat Enterprise Linux 9.

Red Hat Enterprise Linux 9 Configuring and using network file services

A guide to configuring and using network file services in Red Hat Enterprise Linux 9.

Legal Notice

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document describes how to configure and run network file services on Red Hat Enterprise Linux 9, including Samba server and NFS server.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	5
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	6
CHAPTER 1. USING SAMBA AS A SERVER	7
1.1. UNDERSTANDING THE DIFFERENT SAMBA SERVICES AND MODES	7
1.1.1. The Samba services	7
1.1.2. The Samba security services	8
1.1.3. Scenarios when Samba services and Samba client utilities load and reload their configuration	8
1.1.4. Editing the Samba configuration in a safe way	9
1.2. VERIFYING THE SMB.CONF FILE BY USING THE TESTPARM UTILITY	10
1.3. SETTING UP SAMBA AS A STANDALONE SERVER	10
1.3.1. Setting up the server configuration for the standalone server	11
1.3.2. Creating and enabling local user accounts	12
1.4. UNDERSTANDING AND CONFIGURING SAMBA ID MAPPING	12
1.4.1. Planning Samba ID ranges	13
1.4.2. The * default domain	14
1.4.3. Using the tdb ID mapping back end	15
1.4.4. Using the ad ID mapping back end	15
1.4.5. Using the rid ID mapping back end	17
1.4.6. Using the autorid ID mapping back end	19
1.5. SETTING UP SAMBA AS AN AD DOMAIN MEMBER SERVER	21
1.5.1. Joining a RHEL system to an AD domain	21
1.5.2. Using the local authorization plug-in for MIT Kerberos	24
1.6. SETTING UP SAMBA ON AN IDM DOMAIN MEMBER	24
1.6.1. Preparing the IdM domain for installing Samba on domain members	25
1.6.2. Enabling the AES encryption type in Active Directory using a GPO	27
1.6.3. Installing and configuring a Samba server on an IdM client	27
1.6.4. Manually adding an ID mapping configuration if IdM trusts a new domain	29
1.6.5. Additional resources	30
1.7. SETTING UP A SAMBA FILE SHARE THAT USES POSIX ACLS	30
1.7.1. Adding a share that uses POSIX ACLs	31
1.7.2. Setting standard Linux ACLs on a Samba share that uses POSIX ACLs	32
1.7.3. Setting extended ACLs on a Samba share that uses POSIX ACLs	32
1.8. SETTING PERMISSIONS ON A SHARE THAT USES POSIX ACLS	35
1.8.1. Configuring user and group-based share access	35
1.8.2. Configuring host-based share access	35
1.9. SETTING UP A SHARE THAT USES WINDOWS ACLS	36
1.9.1. Granting the SeDiskOperatorPrivilege privilege	36
1.9.2. Enabling Windows ACL support	37
1.9.3. Adding a share that uses Windows ACLs	37
1.9.4. Managing share permissions and file system ACLs of a share that uses Windows ACLs	38
1.10. MANAGING ACLS ON AN SMB SHARE USING SMBACLS	39
1.10.1. Access control entries	39
1.10.2. Displaying ACLs using smbcacls	42
1.10.3. ACE mask calculation	42
1.10.4. Adding, updating, and removing an ACL using smbcacls	43
Adding an ACL	43
Updating an ACL	43
Deleting an ACL	43
1.11. ENABLING USERS TO SHARE DIRECTORIES ON A SAMBA SERVER	43

1.11.1. Enabling the user shares feature	43
1.11.2. Adding a user share	44
1.11.3. Updating settings of a user share	45
1.11.4. Displaying information about existing user shares	45
1.11.5. Listing user shares	46
1.11.6. Deleting a user share	46
1.12. CONFIGURING A SHARE TO ALLOW ACCESS WITHOUT AUTHENTICATION	46
1.12.1. Enabling guest access to a share	47
1.13. CONFIGURING SAMBA FOR MACOS CLIENTS	48
1.13.1. Optimizing the Samba configuration for providing file shares for macOS clients	48
1.14. USING THE SMBCLIENT UTILITY TO ACCESS AN SMB SHARE	49
1.14.1. How the smbclient interactive mode works	49
1.14.2. Using smbclient in interactive mode	50
1.14.3. Using smbclient in scripting mode	50
1.15. SETTING UP SAMBA AS A PRINT SERVER	51
1.15.1. Enabling print server support in Samba	51
1.15.2. Manually sharing specific printers	52
1.16. SETTING UP AUTOMATIC PRINTER DRIVER DOWNLOADS FOR WINDOWS CLIENTS ON SAMBA PRINT SERVERS	53
1.16.1. Basic information about printer drivers	53
Supported driver model version	53
Package-aware drivers	54
Preparing a printer driver for being uploaded	54
Providing 32-bit and 64-bit drivers for a printer to a client	54
1.16.2. Enabling users to upload and preconfigure drivers	54
1.16.3. Setting up the print\$ share	55
1.16.4. Creating a GPO to enable clients to trust the Samba print server	56
1.16.5. Uploading drivers and preconfiguring printers	59
1.17. RUNNING SAMBA ON A SERVER WITH FIPS MODE ENABLED	59
1.17.1. Limitations of using Samba in FIPS mode	59
1.17.2. Using Samba in FIPS mode	60
1.18. TUNING THE PERFORMANCE OF A SAMBA SERVER	60
1.18.1. Setting the SMB protocol version	61
1.18.2. Tuning shares with directories that contain a large number of files	61
1.18.3. Settings that can have a negative performance impact	62
1.19. CONFIGURING SAMBA TO BE COMPATIBLE WITH CLIENTS THAT REQUIRE AN SMB VERSION LOWER THAN THE DEFAULT	62
1.19.1. Setting the minimum SMB protocol version supported by a Samba server	62
1.20. FREQUENTLY USED SAMBA COMMAND-LINE UTILITIES	63
1.20.1. Using the net ads join and net rpc join commands	63
1.20.2. Using the net rpc rights command	64
Listing privileges you can set	64
Granting privileges	65
Revoking privileges	65
1.20.3. Using the net rpc share command	65
Listing shares	65
Adding a share	65
Removing a share	66
1.20.4. Using the net user command	66
Listing domain user accounts	66
Adding a user account to the domain	67
Deleting a user account from the domain	67
1.20.5. Using the rpcclient utility	67

Examples	67
1.20.6. Using the samba-regedit application	68
1.20.7. Using the smbcontrol utility	69
1.20.8. Using the smbpasswd utility	70
1.20.9. Using the smbstatus utility	71
1.20.10. Using the smbtar utility	71
1.20.11. Using the wbinfo utility	72
1.21. ADDITIONAL RESOURCES	73
CHAPTER 2. EXPORTING NFS SHARES	74
2.1. INTRODUCTION TO NFS	74
2.2. SUPPORTED NFS VERSIONS	74
Default NFS version	74
Features of minor NFS versions	74
2.3. THE TCP AND UDP PROTOCOLS IN NFSV3 AND NFSV4	75
2.4. SERVICES REQUIRED BY NFS	75
The RPC services with NFSv4	76
2.5. NFS HOST NAME FORMATS	76
2.6. NFS SERVER CONFIGURATION	77
2.6.1. The /etc/exports configuration file	77
Export entry	77
Default options	78
Default and overridden options	79
2.6.2. The exportfs utility	79
Common exportfs options	79
2.7. NFS AND RPCBIND	80
2.8. INSTALLING NFS	80
2.9. STARTING THE NFS SERVER	80
2.10. TROUBLESHOOTING NFS AND RPCBIND	81
2.11. CONFIGURING THE NFS SERVER TO RUN BEHIND A FIREWALL	82
2.11.1. Configuring the NFSv3-enabled server to run behind a firewall	82
2.11.2. Configuring the NFSv4-only server to run behind a firewall	83
2.11.3. Configuring an NFSv3 client to run behind a firewall	84
2.11.4. Configuring an NFSv4 client to run behind a firewall	85
2.12. EXPORTING RPC QUOTA THROUGH A FIREWALL	85
2.13. ENABLING NFS OVER RDMA (NFSORDMA)	86
2.14. ADDITIONAL RESOURCES	87
CHAPTER 3. SECURING NFS	88
3.1. NFS SECURITY WITH AUTH_SYS AND EXPORT CONTROLS	88
3.2. NFS SECURITY WITH AUTH_GSS	88
3.3. CONFIGURING AN NFS SERVER AND CLIENT TO USE KERBEROS	88
3.4. NFSV4 SECURITY OPTIONS	89
3.5. FILE PERMISSIONS ON MOUNTED NFS EXPORTS	90
CHAPTER 4. ENABLING PNFS SCSI LAYOUTS IN NFS	91
4.1. THE PNFS TECHNOLOGY	91
4.2. PNFS SCSI LAYOUTS	91
Operations between the client and the server	91
Device reservations	91
4.3. CHECKING FOR A SCSI DEVICE COMPATIBLE WITH PNFS	92
4.4. SETTING UP PNFS SCSI ON THE SERVER	93
4.5. SETTING UP PNFS SCSI ON THE CLIENT	93
4.6. RELEASING THE PNFS SCSI RESERVATION ON THE SERVER	94

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting comments on specific passages

1. View the documentation in the **Multi-page HTML** format and ensure that you see the **Feedback** button in the upper right corner after the page fully loads.
2. Use your cursor to highlight the part of the text that you want to comment on.
3. Click the **Add Feedback** button that appears near the highlighted text.
4. Add your feedback and click **Submit**.

Submitting feedback through Jira (account required)

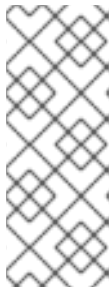
1. Log in to the [Jira](#) website.
2. Click **Create** in the top navigation bar
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Create** at the bottom of the dialogue.

CHAPTER 1. USING SAMBA AS A SERVER

Samba implements the Server Message Block (SMB) protocol in Red Hat Enterprise Linux. The SMB protocol is used to access resources on a server, such as file shares and shared printers. Additionally, Samba implements the Distributed Computing Environment Remote Procedure Call (DCE RPC) protocol used by Microsoft Windows.

You can run Samba as:

- An Active Directory (AD) or NT4 domain member
- A standalone server
- An NT4 Primary Domain Controller (PDC) or Backup Domain Controller (BDC)



NOTE

Red Hat supports the PDC and BDC modes only in existing installations with Windows versions which support NT4 domains. Red Hat recommends not setting up a new Samba NT4 domain, because Microsoft operating systems later than Windows 7 and Windows Server 2008 R2 do not support NT4 domains.

Red Hat does not support running Samba as an AD domain controller (DC).

Independently of the installation mode, you can optionally share directories and printers. This enables Samba to act as a file and print server.

1.1. UNDERSTANDING THE DIFFERENT SAMBA SERVICES AND MODES

This section describes the different services included in Samba and the different modes you can configure.

1.1.1. The Samba services

Samba provides the following services:

smbd

This service provides file sharing and printing services using the SMB protocol. Additionally, the service is responsible for resource locking and for authenticating connecting users. For authenticating domain members, **smbd** requires **winbindd**. The **smb systemd** service starts and stops the **smbd** daemon.

To use the **smbd** service, install the **samba** package.

nmbd

This service provides host name and IP resolution using the NetBIOS over IPv4 protocol. Additionally to the name resolution, the **nmbd** service enables browsing the SMB network to locate domains, work groups, hosts, file shares, and printers. For this, the service either reports this information directly to the broadcasting client or forwards it to a local or master browser. The **nmb systemd** service starts and stops the **nmbd** daemon.

Note that modern SMB networks use DNS to resolve clients and IP addresses. For Kerberos a working DNS setup is required.

To use the **nmbd** service, install the **samba** package.

winbindd

This service provides an interface for the Name Service Switch (NSS) to use AD or NT4 domain users and groups on the local system. This enables, for example, domain users to authenticate to services hosted on a Samba server or to other local services. The **winbind systemd** service starts and stops the **winbindd** daemon.

If you set up Samba as a domain member, **winbindd** must be started before the **smbd** service. Otherwise, domain users and groups are not available to the local system..

To use the **winbindd** service, install the **samba-winbind** package.



IMPORTANT

Red Hat only supports running Samba as a server with the **winbindd** service to provide domain users and groups to the local system. Due to certain limitations, such as missing Windows access control list (ACL) support and NT LAN Manager (NTLM) fallback, SSSD is not supported.

1.1.2. The Samba security services

The **security** parameter in the **[global]** section in the **/etc/samba/smb.conf** file manages how Samba authenticates users that are connecting to the service. Depending on the mode you install Samba in, the parameter must be set to different values:

On an AD domain member, **setsecurity = ads**

In this mode, Samba uses Kerberos to authenticate AD users.

For details about setting up Samba as a domain member, see [Setting up Samba as an AD domain member server](#).

On a standalone server, **setsecurity = user**

In this mode, Samba uses a local database to authenticate connecting users.

For details about setting up Samba as a standalone server, see [Setting up Samba as a standalone server](#).

On an NT4 PDC or BDC, **setsecurity = user**

In this mode, Samba authenticates users to a local or LDAP database.

On an NT4 domain member, **setsecurity = domain**

In this mode, Samba authenticates connecting users to an NT4 PDC or BDC. You cannot use this mode on AD domain members.

For details about setting up Samba as a domain member, see [Setting up Samba as an AD domain member server](#).

Additional resources

- **security** parameter in the **smb.conf(5)** man page

1.1.3. Scenarios when Samba services and Samba client utilities load and reload their configuration

The following describes when Samba services and utilities load and reload their configuration:

- Samba services reload their configuration:
 - Automatically every 3 minutes
 - On manual request, for example, when you run the **smbcontrol all reload-config** command.
- Samba client utilities read their configuration only when you start them.

Note that certain parameters, such as **security** require a restart of the **smb** service to take effect and a reload is not sufficient.

Additional resources

- The **How configuration changes are applied** section in the **smb.conf(5)** man page
- The **smbd(8)**, **nmbd(8)**, and **winbindd(8)** man pages

1.1.4. Editing the Samba configuration in a safe way

Samba services automatically reload their configuration every 3 minutes. This procedure describes how to edit the Samba configuration in a way that prevents the services reload the changes before you have verified the configuration using the **testparm** utility.

Prerequisites

- Samba is installed.

Procedure

1. Create a copy of the **/etc/samba/smb.conf** file:

```
# cp /etc/samba/smb.conf /etc/samba/samba.conf.copy
```

2. Edit the copied file and make the desired changes.
3. Verify the configuration in the **/etc/samba/samba.conf.copy** file:

```
# testparm -s /etc/samba/samba.conf.copy
```

If **testparm** reports errors, fix them and run the command again.

4. Override the **/etc/samba/smb.conf** file with the new configuration:

```
# mv /etc/samba/samba.conf.copy /etc/samba/smb.conf
```

5. Wait until the Samba services automatically reload their configuration or manually reload the configuration:

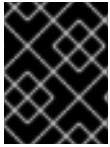
```
# smbcontrol all reload-config
```

Additional resources

- [Scenarios when Samba services and Samba client utilities load and reload their configuration](#)

1.2. VERIFYING THE SMB.CONF FILE BY USING THE TESTPARM UTILITY

The **testparm** utility verifies that the Samba configuration in the `/etc/samba/smb.conf` file is correct. The utility detects invalid parameters and values, but also incorrect settings, such as for ID mapping. If **testparm** reports no problem, the Samba services will successfully load the `/etc/samba/smb.conf` file. Note that **testparm** cannot verify that the configured services will be available or work as expected.



IMPORTANT

Red Hat recommends that you verify the `/etc/samba/smb.conf` file by using **testparm** after each modification of this file.

Prerequisites

- You installed Samba.
- The `/etc/samba/smb.conf` file exists.

Procedure

1. Run the **testparm** utility as the **root** user:

```
# testparm
Load smb config files from /etc/samba/smb.conf
rlimit_max: increasing rlimit_max (1024) to minimum Windows limit (16384)
Unknown parameter encountered: "log levell"
Processing section "[example_share]"
Loaded services file OK.
ERROR: The idmap range for the domain * (tdb) overlaps with the range of DOMAIN (ad)!

Server role: ROLE_DOMAIN_MEMBER

Press enter to see a dump of your service definitions

# Global parameters
[global]
...

[example_share]
...
```

The previous example output reports a non-existent parameter and an incorrect ID mapping configuration.

2. If **testparm** reports incorrect parameters, values, or other errors in the configuration, fix the problem and run the utility again.

1.3. SETTING UP SAMBA AS A STANDALONE SERVER

You can set up Samba as a server that is not a member of a domain. In this installation mode, Samba authenticates users to a local database instead of to a central DC. Additionally, you can enable guest access to allow users to connect to one or multiple services without authentication.

1.3.1. Setting up the server configuration for the standalone server

You can set up the server configuration for a Samba standalone server.

Procedure

1. Install the **samba** package:

```
# dnf install samba
```

2. Edit the **/etc/samba/smb.conf** file and set the following parameters:

```
[global]
workgroup = Example-WG
netbios name = Server
security = user

log file = /var/log/samba/%m.log
log level = 1
```

This configuration defines a standalone server named **Server** within the **Example-WG** work group. Additionally, this configuration enables logging on a minimal level (**1**) and log files will be stored in the **/var/log/samba/** directory. Samba will expand the **%m** macro in the **log file** parameter to the NetBIOS name of connecting clients. This enables individual log files for each client.

3. Optionally, configure file or printer sharing. See:

- [Setting up a share that uses POSIX ACLs](#)
- [Setting up a share that uses Windows ACLs](#)
- [Setting up Samba as a Print Server](#)

4. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

5. If you set up shares that require authentication, create the user accounts.

For details, see [Creating and enabling local user accounts](#).

6. Open the required ports and reload the firewall configuration by using the **firewall-cmd** utility:

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

7. Enable and start the **smb** service:

```
# systemctl enable --now smb
```

Additional resources

- **smb.conf(5)** man page

1.3.2. Creating and enabling local user accounts

To enable users to authenticate when they connect to a share, you must create the accounts on the Samba host both in the operating system and in the Samba database. Samba requires the operating system account to validate the Access Control Lists (ACL) on file system objects and the Samba account to authenticate connecting users.

If you use the **passdb backend = tdbsam** default setting, Samba stores user accounts in the **/var/lib/samba/private/passdb.tdb** database.

You can create a local Samba user named **example**.

Prerequisites

- Samba is installed and configured as a standalone server.

Procedure

1. Create the operating system account:

```
# useradd -M -s /sbin/nologin example
```

This command adds the **example** account without creating a home directory. If the account is only used to authenticate to Samba, assign the **/sbin/nologin** command as shell to prevent the account from logging in locally.

2. Set a password to the operating system account to enable it:

```
# passwd example
Enter new UNIX password: password
Retype new UNIX password: password
passwd: password updated successfully
```

Samba does not use the password set on the operating system account to authenticate. However, you need to set a password to enable the account. If an account is disabled, Samba denies access if this user connects.

3. Add the user to the Samba database and set a password to the account:

```
# smbpasswd -a example
New SMB password: password
Retype new SMB password: password
Added user example.
```

Use this password to authenticate when using this account to connect to a Samba share.

4. Enable the Samba account:

```
# smbpasswd -e example
Enabled user example.
```

1.4. UNDERSTANDING AND CONFIGURING SAMBA ID MAPPING

Windows domains distinguish users and groups by unique Security Identifiers (SID). However, Linux

requires unique UIDs and GIDs for each user and group. If you run Samba as a domain member, the **winbindd** service is responsible for providing information about domain users and groups to the operating system.

To enable the **winbindd** service to provide unique IDs for users and groups to Linux, you must configure ID mapping in the **/etc/samba/smb.conf** file for:

- The local database (default domain)
- The AD or NT4 domain the Samba server is a member of
- Each trusted domain from which users must be able to access resources on this Samba server

Samba provides different ID mapping back ends for specific configurations. The most frequently used back ends are:

Back end	Use case
tdb	The * default domain only
ad	AD domains only
rid	AD and NT4 domains
autorid	AD, NT4, and the * default domain

1.4.1. Planning Samba ID ranges

Regardless of whether you store the Linux UIDs and GIDs in AD or if you configure Samba to generate them, each domain configuration requires a unique ID range that must not overlap with any of the other domains.



WARNING

If you set overlapping ID ranges, Samba fails to work correctly.

Example 1.1. Unique ID Ranges

The following shows non-overlapping ID mapping ranges for the default (*), **AD-DOM**, and the **TRUST-DOM** domains.

```
[global]
...
idmap config * : backend = tdb
idmap config * : range = 10000-999999

idmap config AD-DOM:backend = rid
idmap config AD-DOM:range = 2000000-2999999
```

```
idmap config TRUST-DOM:backend = rid
idmap config TRUST-DOM:range = 4000000-4999999
```



IMPORTANT

You can only assign one range per domain. Therefore, leave enough space between the domains ranges. This enables you to extend the range later if your domain grows.

If you later assign a different range to a domain, the ownership of files and directories previously created by these users and groups will be lost.

1.4.2. The * default domain

In a domain environment, you add one ID mapping configuration for each of the following:

- The domain the Samba server is a member of
- Each trusted domain that should be able to access the Samba server

However, for all other objects, Samba assigns IDs from the default domain. This includes:

- Local Samba users and groups
- Samba built-in accounts and groups, such as **BUILTIN\Administrators**



IMPORTANT

You must configure the default domain as described in this section to enable Samba to operate correctly.

The default domain back end must be writable to permanently store the assigned IDs.

For the default domain, you can use one of the following back ends:

tdb

When you configure the default domain to use the **tdb** back end, set an ID range that is big enough to include objects that will be created in the future and that are not part of a defined domain ID mapping configuration.

For example, set the following in the **[global]** section in the **/etc/samba/smb.conf** file:

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

For further details, see [Using the TDB ID mapping back end](#) .

autorid

When you configure the default domain to use the **autorid** back end, adding additional ID mapping configurations for domains is optional.

For example, set the following in the **[global]** section in the **/etc/samba/smb.conf** file:

```
idmap config * : backend = autorid
idmap config * : range = 10000-999999
```

For further details, see [Using the autorid ID mapping back end](#).

1.4.3. Using the tdb ID mapping back end

The **winbindd** service uses the writable **tdb** ID mapping back end by default to store Security Identifier (SID), UID, and GID mapping tables. This includes local users, groups, and built-in principals.

Use this back end only for the * default domain. For example:

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

Additional resources

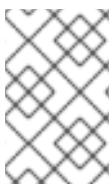
- [The * default domain](#).

1.4.4. Using the ad ID mapping back end

You can configure a Samba AD member to use the **ad** ID mapping back end.

The **ad** ID mapping back end implements a read-only API to read account and group information from AD. This provides the following benefits:

- All user and group settings are stored centrally in AD.
- User and group IDs are consistent on all Samba servers that use this back end.
- The IDs are not stored in a local database which can corrupt, and therefore file ownerships cannot be lost.



NOTE

The **ad** ID mapping back end does not support Active Directory domains with one-way trusts. If you configure a domain member in an Active Directory with one-way trusts, use instead one of the following ID mapping back ends: **tdb**, **rid**, or **autorid**.

The ad back end reads the following attributes from AD:

AD attribute name	Object type	Mapped to
sAMAccountName	User and group	User or group name, depending on the object
uidNumber	User	User ID (UID)
gidNumber	Group	Group ID (GID)
loginShell ^[a]	User	Path to the shell of the user

AD attribute name	Object type	Mapped to
unixHomeDirectory [a]	User	Path to the home directory of the user
primaryGroupID [b]	User	Primary group ID
[a] Samba only reads this attribute if you set idmap config DOMAIN:unix_nss_info = yes .		
[b] Samba only reads this attribute if you set idmap config DOMAIN:unix_primary_group = yes .		

Prerequisites

- Both users and groups must have unique IDs set in AD, and the IDs must be within the range configured in the **/etc/samba/smb.conf** file. Objects whose IDs are outside of the range will not be available on the Samba server.
- Users and groups must have all required attributes set in AD. If required attributes are missing, the user or group will not be available on the Samba server. The required attributes depend on your configuration. .Prerequisites
- You installed Samba.
- The Samba configuration, except ID mapping, exists in the **/etc/samba/smb.conf** file.

Procedure

1. Edit the **[global]** section in the **/etc/samba/smb.conf** file:
 - a. Add an ID mapping configuration for the default domain (*) if it does not exist. For example:

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

- b. Enable the **ad** ID mapping back end for the AD domain:

```
idmap config DOMAIN : backend = ad
```

- c. Set the range of IDs that is assigned to users and groups in the AD domain. For example:

```
idmap config DOMAIN : range = 2000000-2999999
```



IMPORTANT

The range must not overlap with any other domain configuration on this server. Additionally, the range must be set big enough to include all IDs assigned in the future. For further details, see [Planning Samba ID ranges](#).

- d. Set that Samba uses the [RFC 2307](#) schema when reading attributes from AD:

```
idmap config DOMAIN : schema_mode = rfc2307
```

- e. To enable Samba to read the login shell and the path to the users home directory from the corresponding AD attribute, set:

```
idmap config DOMAIN : unix_nss_info = yes
```

Alternatively, you can set a uniform domain-wide home directory path and login shell that is applied to all users. For example:

```
template shell = /bin/bash
template homedir = /home/%U
```

- f. By default, Samba uses the **primaryGroupID** attribute of a user object as the user's primary group on Linux. Alternatively, you can configure Samba to use the value set in the **gidNumber** attribute instead:

```
idmap config DOMAIN : unix_primary_group = yes
```

2. Verify the `/etc/samba/smb.conf` file:

```
# testparm
```

3. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

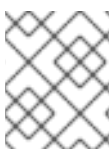
Additional resources

- [The * default domain](#)
- **smb.conf(5)** and **idmap_ad(8)** man pages
- **VARIABLE SUBSTITUTIONS** section in the **smb.conf(5)** man page

1.4.5. Using the rid ID mapping back end

You can configure a Samba domain member to use the **rid** ID mapping back end.

Samba can use the relative identifier (RID) of a Windows SID to generate an ID on Red Hat Enterprise Linux.



NOTE

The RID is the last part of a SID. For example, if the SID of a user is **S-1-5-21-5421822485-1151247151-421485315-30014**, then **30014** is the corresponding RID.

The **rid** ID mapping back end implements a read-only API to calculate account and group information based on an algorithmic mapping scheme for AD and NT4 domains. When you configure the back end, you must set the lowest and highest RID in the **idmap config *DOMAIN* : range** parameter. Samba will not map users or groups with a lower or higher RID than set in this parameter.



IMPORTANT

As a read-only back end, **rid** cannot assign new IDs, such as for **BUILTIN** groups. Therefore, do not use this back end for the ***** default domain.

Benefits of using the rid back end

- All domain users and groups that have an RID within the configured range are automatically available on the domain member.
- You do not need to manually assign IDs, home directories, and login shells.

Drawbacks of using the rid back end

- All domain users get the same login shell and home directory assigned. However, you can use variables.
- User and group IDs are only the same across Samba domain members if all use the **rid** back end with the same ID range settings.
- You cannot exclude individual users or groups from being available on the domain member. Only users and groups outside of the configured range are excluded.
- Based on the formulas the **winbindd** service uses to calculate the IDs, duplicate IDs can occur in multi-domain environments if objects in different domains have the same RID.

Prerequisites

- You installed Samba.
- The Samba configuration, except ID mapping, exists in the **/etc/samba/smb.conf** file.

Procedure

1. Edit the **[global]** section in the **/etc/samba/smb.conf** file:
 - a. Add an ID mapping configuration for the default domain (*****) if it does not exist. For example:

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

- b. Enable the **rid** ID mapping back end for the domain:

```
idmap config DOMAIN : backend = rid
```

- c. Set a range that is big enough to include all RIDs that will be assigned in the future. For example:

```
idmap config DOMAIN : range = 2000000-2999999
```

Samba ignores users and groups whose RIDs in this domain are not within the range.



IMPORTANT

The range must not overlap with any other domain configuration on this server. Additionally, the range must be set big enough to include all IDs assigned in the future. For further details, see [Planning Samba ID ranges](#).

- d. Set a shell and home directory path that will be assigned to all mapped users. For example:

```
template shell = /bin/bash
template homedir = /home/%U
```

2. Verify the `/etc/samba/smb.conf` file:

```
# testparm
```

3. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

Additional resources

- [The * default domain](#)
- **VARIABLE SUBSTITUTIONS** section in the `smb.conf(5)` man page
- Calculation of the local ID from a RID, see the `idmap_rid(8)` man page

1.4.6. Using the autorid ID mapping back end

You can configure a Samba domain member to use the **autorid** ID mapping back end.

The **autorid** back end works similar to the **rid** ID mapping back end, but can automatically assign IDs for different domains. This enables you to use the **autorid** back end in the following situations:

- Only for the * default domain
- For the * default domain and additional domains, without the need to create ID mapping configurations for each of the additional domains
- Only for specific domains



NOTE

If you use **autorid** for the default domain, adding additional ID mapping configuration for domains is optional.

Parts of this section were adopted from the [idmap config autorid](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

Benefits of using the autorid back end

- All domain users and groups whose calculated UID and GID is within the configured range are automatically available on the domain member.

- You do not need to manually assign IDs, home directories, and login shells.
- No duplicate IDs, even if multiple objects in a multi-domain environment have the same RID.

Drawbacks

- User and group IDs are not the same across Samba domain members.
- All domain users get the same login shell and home directory assigned. However, you can use variables.
- You cannot exclude individual users or groups from being available on the domain member. Only users and groups whose calculated UID or GID is outside of the configured range are excluded.

Prerequisites

- You installed Samba.
- The Samba configuration, except ID mapping, exists in the **/etc/samba/smb.conf** file.

Procedure

1. Edit the **[global]** section in the **/etc/samba/smb.conf** file:

- a. Enable the **autorid** ID mapping back end for the ***** default domain:

```
idmap config * : backend = autorid
```

- b. Set a range that is big enough to assign IDs for all existing and future objects. For example:

```
idmap config * : range = 10000-999999
```

Samba ignores users and groups whose calculated IDs in this domain are not within the range.



WARNING

After you set the range and Samba starts using it, you can only increase the upper limit of the range. Any other change to the range can result in new ID assignments, and thus in losing file ownerships.

- c. Optionally, set a range size. For example:

```
idmap config * : rangesize = 200000
```

Samba assigns this number of continuous IDs for each domain's object until all IDs from the range set in the **idmap config * : range** parameter are taken.

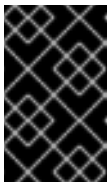
**NOTE**

If you set a `rangesize`, you need to adapt the range accordingly. The range needs to be a multiple of the `rangesize`.

- d. Set a shell and home directory path that will be assigned to all mapped users. For example:

```
template shell = /bin/bash
template homedir = /home/%U
```

- e. Optionally, add additional ID mapping configuration for domains. If no configuration for an individual domain is available, Samba calculates the ID using the **autorid** back end settings in the previously configured * default domain.

**IMPORTANT**

The range must not overlap with any other domain configuration on this server. Additionally, the range must be set big enough to include all IDs assigned in the future. For further details, see [Planning Samba ID ranges](#).

2. Verify the `/etc/samba/smb.conf` file:

```
# testparm
```

3. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

Additional resources

- **THE MAPPING FORMULAS** section in the `idmap_autorid(8)` man page
- **rangesize** parameter description in the `idmap_autorid(8)` man page
- **VARIABLE SUBSTITUTIONS** section in the `smb.conf(5)` man page

1.5. SETTING UP SAMBA AS AN AD DOMAIN MEMBER SERVER

If you are running an AD or NT4 domain, use Samba to add your Red Hat Enterprise Linux server as a member to the domain to gain the following:

- Access domain resources on other domain members
- Authenticate domain users to local services, such as **sshd**
- Share directories and printers hosted on the server to act as a file and print server

1.5.1. Joining a RHEL system to an AD domain

Samba Winbind is an alternative to the System Security Services Daemon (SSSD) for connecting a Red Hat Enterprise Linux (RHEL) system with Active Directory (AD). You can join a RHEL system to an AD domain by using **realmd** to configure Samba Winbind.

Procedure

1. If your AD requires the deprecated RC4 encryption type for Kerberos authentication, enable support for these ciphers in RHEL:

```
# update-crypto-policies --set DEFAULT:AD-SUPPORT
```

2. Install the following packages:

```
# dnf install realmd oddjob-mkhomedir oddjob samba-winbind-clients \
    samba-winbind samba-common-tools samba-winbind-krb5-locator
```

3. To share directories or printers on the domain member, install the **samba** package:

```
# dnf install samba
```

4. Backup the existing **/etc/samba/smb.conf** Samba configuration file:

```
# mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
```

5. Join the domain. For example, to join a domain named **ad.example.com**:

```
# realm join --membership-software=samba --client-software=winbind ad.example.com
```

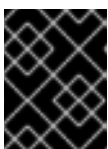
Using the previous command, the **realm** utility automatically:

- Creates a **/etc/samba/smb.conf** file for a membership in the **ad.example.com** domain
 - Adds the **winbind** module for user and group lookups to the **/etc/nsswitch.conf** file
 - Updates the Pluggable Authentication Module (PAM) configuration files in the **/etc/pam.d/** directory
 - Starts the **winbind** service and enables the service to start when the system boots
6. Optionally, set an alternative ID mapping back end or customized ID mapping settings in the **/etc/samba/smb.conf** file.

For details, see [Understanding and configuring Samba ID mapping](#).

1. Verify that the **winbind** service is running:

```
# systemctl status winbind
...
Active: active (running) since Tue 2018-11-06 19:10:40 CET; 15s ago
```



IMPORTANT

To enable Samba to query domain user and group information, the **winbind** service must be running before you start **smb**.

2. If you installed the **samba** package to share directories and printers, enable and start the **smb** service:

■

```
# systemctl enable --now smb
```

- Optionally, if you are authenticating local logins to Active Directory, enable the **winbind_krb5_localauth** plug-in. See [Using the local authorization plug-in for MIT Kerberos](#).

Verification steps

- Display an AD user's details, such as the AD administrator account in the AD domain:

```
# getent passwd "AD\administrator"
AD\administrator:*:10000:10000::/home/administrator@AD:/bin/bash
```

- Query the members of the domain users group in the AD domain:

```
# getent group "AD\Domain Users"
AD\domain users:x:10000:user1,user2
```

- Optionally, verify that you can use domain users and groups when you set permissions on files and directories. For example, to set the owner of the **/srv/samba/example.txt** file to **AD\administrator** and the group to **AD\Domain Users**:

```
# chown "AD\administrator":"AD\Domain Users" /srv/samba/example.txt
```

- Verify that Kerberos authentication works as expected:
 - On the AD domain member, obtain a ticket for the **administrator@AD.EXAMPLE.COM** principal:

```
# kinit administrator@AD.EXAMPLE.COM
```

- Display the cached Kerberos ticket:

```
# klist
Ticket cache: KCM:0
Default principal: administrator@AD.EXAMPLE.COM

Valid starting    Expires          Service principal
01.11.2018 10:00:00 01.11.2018 20:00:00
krbtgt/AD.EXAMPLE.COM@AD.EXAMPLE.COM
renew until 08.11.2018 05:00:00
```

- Display the available domains:

```
# wbinfo --all-domains
BUILTIN
SAMBA-SERVER
AD
```

Additional resources

- If you do not want to use the deprecated RC4 ciphers, you can enable the AES encryption type in AD. See

- [Enabling the AES encryption type in Active Directory using a GPO](#) . Note that this can have an impact on other services in your AD.
- **realm(8)** man page

1.5.2. Using the local authorization plug-in for MIT Kerberos

The **winbind** service provides Active Directory users to the domain member. In certain situations, administrators want to enable domain users to authenticate to local services, such as an SSH server, which are running on the domain member. When using Kerberos to authenticate the domain users, enable the **winbind_krb5_localauth** plug-in to correctly map Kerberos principals to Active Directory accounts through the **winbind** service.

For example, if the **sAMAccountName** attribute of an Active Directory user is set to **EXAMPLE** and the user tries to log with the user name lowercase, Kerberos returns the user name in upper case. As a consequence, the entries do not match and authentication fails.

Using the **winbind_krb5_localauth** plug-in, the account names are mapped correctly. Note that this only applies to GSSAPI authentication and not for getting the initial ticket granting ticket (TGT).

Prerequisites

- Samba is configured as a member of an Active Directory.
- Red Hat Enterprise Linux authenticates log in attempts against Active Directory.
- The **winbind** service is running.

Procedure

Edit the **/etc/krb5.conf** file and add the following section:

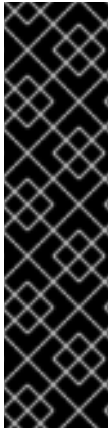
```
[plugins]
localauth = {
    module = winbind:/usr/lib64/samba/krb5/winbind_krb5_localauth.so
    enable_only = winbind
}
```

Additional resources

- **winbind_krb5_localauth(8)** man page.

1.6. SETTING UP SAMBA ON AN IDM DOMAIN MEMBER

You can set up Samba on a host that is joined to a Red Hat Identity Management (IdM) domain. Users from IdM and also, if available, from trusted Active Directory (AD) domains, can access shares and printer services provided by Samba.



IMPORTANT

Using Samba on an IdM domain member is an unsupported Technology Preview feature and contains certain limitations. For example, IdM trust controllers do not support the Active Directory Global Catalog service, and they do not support resolving IdM groups using the Distributed Computing Environment / Remote Procedure Calls (DCE/RPC) protocols. As a consequence, AD users can only access Samba shares and printers hosted on IdM clients when logged in to other IdM clients; AD users logged into a Windows machine can not access Samba shares hosted on an IdM domain member.

Customers deploying Samba on IdM domain members are encouraged to provide feedback to Red Hat.

Prerequisites

- The host is joined as a client to the IdM domain.
- Both the IdM servers and the client must run on RHEL 9.0 or later.

1.6.1. Preparing the IdM domain for installing Samba on domain members

Before you can set up Samba on an IdM client, you must prepare the IdM domain using the **ipa-adtrust-install** utility on an IdM server.



NOTE

Any system where you run the **ipa-adtrust-install** command automatically becomes an AD trust controller. However, you must run **ipa-adtrust-install** only once on an IdM server.

Prerequisites

- IdM server is installed.
- You need root privileges to install packages and restart IdM services.

Procedure

1. Install the required packages:

```
[root@ipaserver ~]# dnf install ipa-server-trust-ad samba-client
```

2. Authenticate as the IdM administrative user:

```
[root@ipaserver ~]# kinit admin
```

3. Run the **ipa-adtrust-install** utility:

```
[root@ipaserver ~]# ipa-adtrust-install
```

The DNS service records are created automatically if IdM was installed with an integrated DNS server.

If you installed IdM without an integrated DNS server, **ipa-adtrust-install** prints a list of service records that you must manually add to DNS before you can continue.

4. The script prompts you that the **/etc/samba/smb.conf** already exists and will be rewritten:

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing Samba configuration.
```

```
Do you wish to continue? [no]: yes
```

5. The script prompts you to configure the **slapi-nis** plug-in, a compatibility plug-in that allows older Linux clients to work with trusted users:

```
Do you want to enable support for trusted domains in Schema Compatibility plugin?
This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.
```

```
Enable trusted domains support in slapi-nis? [no]: yes
```

6. When prompted, enter the NetBIOS name for the IdM domain or press **Enter** to accept the name suggested:

```
Trust is configured but no NetBIOS domain name found, setting it now.
Enter the NetBIOS name for the IPA domain.
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.
Example: EXAMPLE.
```

```
NetBIOS domain name [IDM]:
```

7. You are prompted to run the SID generation task to create a SID for any existing users:

```
Do you want to run the ipa-sidgen task? [no]: yes
```

This is a resource-intensive task, so if you have a high number of users, you can run this at another time.

8. **(Optional)** By default, the Dynamic RPC port range is defined as **49152-65535** for Windows Server 2008 and later. If you need to define a different Dynamic RPC port range for your environment, configure Samba to use different ports and open those ports in your firewall settings. The following example sets the port range to **55000-65000**.

```
[root@ipaserver ~]# net conf setparm global 'rpc server dynamic port range' 55000-65000
[root@ipaserver ~]# firewall-cmd --add-port=55000-65000/tcp
[root@ipaserver ~]# firewall-cmd --runtime-to-permanent
```

9. Restart the **ipa** service:

```
[root@ipaserver ~]# ipactl restart
```

10. Use the **smbclient** utility to verify that Samba responds to Kerberos authentication from the IdM side:

```
[root@ipaserver ~]# smbclient -L server.idm.example.com -U user_name --use-kerberos=required
```

```
lp_load_ex: changing to config backend registry
Sharename      Type      Comment
-----
IPC$           IPC       IPC Service (Samba 4.15.2)
...
```

1.6.2. Enabling the AES encryption type in Active Directory using a GPO

This section describes how to enable the AES encryption type in Active Directory (AD) using a group policy object (GPO). Certain features on RHEL, such as running a Samba server on an IdM client, require this encryption type.

Note that RHEL no longer supports the weak DES and RC4 encryption types.

Prerequisites

- You are logged into AD as a user who can edit group policies.
- The **Group Policy Management Console** is installed on the computer.

Procedure

1. Open the **Group Policy Management Console**.
2. Right-click **Default Domain Policy**, and select **Edit**. The **Group Policy Management Editor** opens.
3. Navigate to **Computer Configuration** → **Policies** → **Windows Settings** → **Security Settings** → **Local Policies** → **Security Options**.
4. Double-click the **Network security: Configure encryption types allowed for Kerberos** policy.
5. Select **AES256_HMAC_SHA1** and, optionally, **Future encryption types**.
6. Click **OK**.
7. Close the **Group Policy Management Editor**.
8. Repeat the steps for the **Default Domain Controller Policy**.
9. Wait until the Windows domain controllers (DC) applied the group policy automatically. Alternatively, to apply the GPO manually on a DC, enter the following command using an account that has administrator permissions:

```
C:\> gpupdate /force /target:computer
```

1.6.3. Installing and configuring a Samba server on an IdM client

You can install and configure Samba on a client enrolled in an IdM domain.

Prerequisites

- Both the IdM servers and the client must run on RHEL 9.0 or later.

- The IdM domain is prepared as described in [Preparing the IdM domain for installing Samba on domain members](#).
- If IdM has a trust configured with AD, enable the AES encryption type for Kerberos. For example, use a group policy object (GPO) to enable the AES encryption type. For details, see [Enabling AES encryption in Active Directory using a GPO](#).

Procedure

1. Install the **ipa-client-samba** package:

```
[root@idm_client]# dnf install ipa-client-samba
```

2. Use the **ipa-client-samba** utility to prepare the client and create an initial Samba configuration:

```
[root@idm_client]# ipa-client-samba
Searching for IPA server...
IPA server: DNS discovery
Chosen IPA master: idm_server.idm.example.com
SMB principal to be created: cifs/idm_client.idm.example.com@IDM.EXAMPLE.COM
NetBIOS name to be used: IDM_CLIENT
Discovered domains to use:

Domain name: idm.example.com
NetBIOS name: IDM
SID: S-1-5-21-525930803-952335037-206501584
ID range: 212000000 - 212199999

Domain name: ad.example.com
NetBIOS name: AD
SID: None
ID range: 1918400000 - 1918599999

Continue to configure the system with these values? [no]: yes
Samba domain member is configured. Please check configuration at /etc/samba/smb.conf
and start smb and winbind services
```

3. By default, **ipa-client-samba** automatically adds the **[homes]** section to the **/etc/samba/smb.conf** file that dynamically shares a user's home directory when the user connects. If users do not have home directories on this server, or if you do not want to share them, remove the following lines from **/etc/samba/smb.conf**:

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

4. Share directories and printers. For details, see:

- [Setting up a Samba file share that uses POSIX ACLs](#)
- [Setting up a share that uses Windows ACLs](#)
- [Setting up Samba as a print server](#)

5. Open the ports required for a Samba client in the local firewall:


```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

6. Enable and start the **smb** and **winbind** services:

```
[root@idm_client]# systemctl enable --now smb winbind
```

Verification steps

Run the following verification step on a different IdM domain member that has the **samba-client** package installed:

- List the shares on the Samba server using Kerberos authentication:

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

Sharename      Type      Comment
-----
example        Disk
IPC$           IPC       IPC Service (Samba 4.15.2)
...
```

Additional resources

- **ipa-client-samba(1)** man page

1.6.4. Manually adding an ID mapping configuration if IdM trusts a new domain

Samba requires an ID mapping configuration for each domain from which users access resources. On an existing Samba server running on an IdM client, you must manually add an ID mapping configuration after the administrator added a new trust to an Active Directory (AD) domain.

Prerequisites

- You configured Samba on an IdM client. Afterward, a new trust was added to IdM.
- The DES and RC4 encryption types for Kerberos must be disabled in the trusted AD domain. For security reasons, RHEL 9 does not support these weak encryption types.

Procedure

1. Authenticate using the host's keytab:

```
[root@idm_client]# kinit -k
```

2. Use the **ipa idrange-find** command to display both the base ID and the ID range size of the new domain. For example, the following command displays the values for the **ad.example.com** domain:

```
[root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw
-----
1 range matched
-----
```

```

cn: AD.EXAMPLE.COM_id_range
ipabaseid: 1918400000
ipaidrangesize: 200000
ipabaserid: 0
ipanttrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
iparangetype: ipa-ad-trust
-----

```

```

Number of entries returned 1
-----

```

You need the values from the **ipabaseid** and **ipaidrangesize** attributes in the next steps.

- To calculate the highest usable ID, use the following formula:

```
maximum_range = ipabaseid + ipaidrangesize - 1
```

With the values from the previous step, the highest usable ID for the **ad.example.com** domain is **1918599999** ($1918400000 + 200000 - 1$).

- Edit the **/etc/samba/smb.conf** file, and add the ID mapping configuration for the domain to the **[global]** section:

```

idmap config AD : range = 1918400000 - 1918599999
idmap config AD : backend = sss

```

Specify the value from **ipabaseid** attribute as the lowest and the computed value from the previous step as the highest value of the range.

- Restart the **smb** and **winbind** services:

```
[root@idm_client]# systemctl restart smb winbind
```

Verification steps

- List the shares on the Samba server using Kerberos authentication:

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry
```

Sharename	Type	Comment
example	Disk	
IPC\$	IPC	IPC Service (Samba 4.15.2)
...		

1.6.5. Additional resources

- [Installing an Identity Management client](#)

1.7. SETTING UP A SAMBA FILE SHARE THAT USES POSIX ACLS

As a Linux service, Samba supports shares with POSIX ACLs. They enable you to manage permissions locally on the Samba server using utilities, such as **chmod**. If the share is stored on a file system that supports extended attributes, you can define ACLs with multiple users and groups.

**NOTE**

If you need to use fine-granular Windows ACLs instead, see [Setting up a share that uses Windows ACLs](#).

Parts of this section were adopted from the [Setting up a Share Using POSIX ACLs](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

1.7.1. Adding a share that uses POSIX ACLs

You can create a share named **example** that provides the content of the `/srv/samba/example/` directory, and uses POSIX ACLs.

Prerequisites

Samba has been set up in one of the following modes:

- [Standalone server](#)
- [Domain member](#)

Procedure

1. Create the folder if it does not exist. For example:

```
# mkdir -p /srv/samba/example/
```

2. If you run SELinux in **enforcing** mode, set the **samba_share_t** context on the directory:

```
# semanage fcontext -a -t samba_share_t "/srv/samba/example(/.*)?"
# restorecon -Rv /srv/samba/example/
```

3. Set file system ACLs on the directory. For details, see:
 - [Setting standard ACLs on a Samba Share that uses POSIX ACLs](#)
 - [Setting extended ACLs on a share that uses POSIX ACLs](#) .
4. Add the example share to the `/etc/samba/smb.conf` file. For example, to add the share write-enabled:

```
[example]
path = /srv/samba/example/
read only = no
```

**NOTE**

Regardless of the file system ACLs; if you do not set **read only = no**, Samba shares the directory in read-only mode.

5. Verify the `/etc/samba/smb.conf` file:

```
# testparm
```

- Open the required ports and reload the firewall configuration using the **firewall-cmd** utility:

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

- Restart the **smb** service:

```
# systemctl restart smb
```

1.7.2. Setting standard Linux ACLs on a Samba share that uses POSIX ACLs

The standard ACLs on Linux support setting permissions for one owner, one group, and for all other undefined users. You can use the **chown**, **chgrp**, and **chmod** utility to update the ACLs. If you require precise control, then you use the more complex POSIX ACLs, see

[Setting extended ACLs on a Samba share that uses POSIX ACLs](#) .

The following procedure sets the owner of the **/srv/samba/example/** directory to the **root** user, grants read and write permissions to the **Domain Users** group, and denies access to all other users.

Prerequisites

- The Samba share on which you want to set the ACLs exists.

Procedure

```
# chown root:"Domain Users" /srv/samba/example/
# chmod 2770 /srv/samba/example/
```



NOTE

Enabling the set-group-ID (SGID) bit on a directory automatically sets the default group for all new files and subdirectories to that of the directory group, instead of the usual behavior of setting it to the primary group of the user who created the new directory entry.

Additional resources

- chown(1)** and **chmod(1)** man pages

1.7.3. Setting extended ACLs on a Samba share that uses POSIX ACLs

If the file system the shared directory is stored on supports extended ACLs, you can use them to set complex permissions. Extended ACLs can contain permissions for multiple users and groups.

Extended POSIX ACLs enable you to configure complex ACLs with multiple users and groups. However, you can only set the following permissions:

- No access
- Read access
- Write access

- Full control

If you require the fine-granular Windows permissions, such as **Create folder / append data**, configure the share to use Windows ACLs.

See [Setting up a share that uses Windows ACLs](#).

The following procedure shows how to enable extended ACLs on a share. Additionally, it contains an example about setting extended ACLs.

Prerequisites

- The Samba share on which you want to set the ACLs exists.

Procedure

1. Enable the following parameter in the share's section in the `/etc/samba/smb.conf` file to enable ACL inheritance of extended ACLs:

```
inherit acls = yes
```

For details, see the parameter description in the `smb.conf(5)` man page.

2. Restart the **smb** service:

```
# systemctl restart smb
```

3. Set the ACLs on the directory. For example:

Example 1.2. Setting Extended ACLs

The following procedure sets read, write, and execute permissions for the **Domain Admins** group, read, and execute permissions for the **Domain Users** group, and deny access to everyone else on the `/srv/samba/example/` directory:

1. Disable auto-granting permissions to the primary group of user accounts:

```
# setfacl -m group::--- /srv/samba/example/
# setfacl -m default:group::--- /srv/samba/example/
```

The primary group of the directory is additionally mapped to the dynamic **CREATOR GROUP** principal. When you use extended POSIX ACLs on a Samba share, this principal is automatically added and you cannot remove it.

2. Set the permissions on the directory:

- a. Grant read, write, and execute permissions to the **Domain Admins** group:

```
# setfacl -m group:"DOMAINDomain Admins":rwx /srv/samba/example/
```

- b. Grant read and execute permissions to the **Domain Users** group:

```
# setfacl -m group:"DOMAINDomain Users":r-x /srv/samba/example/
```

- c. Set permissions for the **other** ACL entry to deny access to users that do not match the other ACL entries:

```
# setfacl -R -m other::--- /srv/samba/example/
```

These settings apply only to this directory. In Windows, these ACLs are mapped to the **This folder only** mode.

- 3. To enable the permissions set in the previous step to be inherited by new file system objects created in this directory:

```
# setfacl -m default:group:"DOMAIN\Domain Admins":rwx /srv/samba/example/  
# setfacl -m default:group:"DOMAIN\Domain Users":r-x /srv/samba/example/  
# setfacl -m default:other::--- /srv/samba/example/
```

With these settings, the **This folder only** mode for the principals is now set to **This folder, subfolders, and files**.

Samba maps the permissions set in the procedure to the following Windows ACLs:

Principal	Access	Applies to
Domain\Domain Admins	Full control	This folder, subfolders, and files
Domain\Domain Users	Read & execute	This folder, subfolders, and files
Everyone ^[a]	None	This folder, subfolders, and files
owner (Unix User\owner) ^[b]	Full control	This folder only
primary_group (Unix User\primary_group) ^[c]	None	This folder only
CREATOR OWNER ^[d] ^[e]	Full control	Subfolders and files only
CREATOR GROUP ^[e] ^[f]	None	Subfolders and files only

[a] Samba maps the permissions for this principal from the **other** ACL entry.

[b] Samba maps the owner of the directory to this entry.

[c] Samba maps the primary group of the directory to this entry.

[d] On new file system objects, the creator inherits automatically the permissions of this principal.

[e] Configuring or removing these principals from the ACLs not supported on shares that use POSIX ACLs.

[f] On new file system objects, the creator’s primary group inherits automatically the permissions of this principal.

1.8. SETTING PERMISSIONS ON A SHARE THAT USES POSIX ACLS

Optionally, to limit or grant access to a Samba share, you can set certain parameters in the share's section in the **/etc/samba/smb.conf** file.



NOTE

Share-based permissions manage if a user, group, or host is able to access a share. These settings do not affect file system ACLs.

Use share-based settings to restrict access to shares, for example, to deny access from specific hosts.

Prerequisites

- A share with POSIX ACLs has been set up.

1.8.1. Configuring user and group-based share access

User and group-based access control enables you to grant or deny access to a share for certain users and groups.

Prerequisites

- The Samba share on which you want to set user or group-based access exists.

Procedure

1. For example, to enable all members of the **Domain Users** group to access a share while access is denied for the **user** account, add the following parameters to the share's configuration:

```
valid users = +DOMAIN\Domain Users"
invalid users = DOMAINuser
```

The **invalid users** parameter has a higher priority than the **valid users** parameter. For example, if the **user** account is a member of the **Domain Users** group, access is denied to this account when you use the previous example.

2. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

Additional resources

- **smb.conf(5)** man page

1.8.2. Configuring host-based share access

Host-based access control enables you to grant or deny access to a share based on client's host names, IP addresses, or IP range.

The following procedure explains how to enable the **127.0.0.1** IP address, the **192.0.2.0/24** IP range, and the **client1.example.com** host to access a share, and additionally deny access for the **client2.example.com** host:

Prerequisites

- The Samba share on which you want to set host-based access exists.

Procedure

1. Add the following parameters to the configuration of the share in the **/etc/samba/smb.conf** file:

```
hosts allow = 127.0.0.1 192.0.2.0/24 client1.example.com
hosts deny = client2.example.com
```

The **hosts deny** parameter has a higher priority than **hosts allow**. For example, if **client1.example.com** resolves to an IP address that is listed in the **hosts allow** parameter, access for this host is denied.

2. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

Additional resources

- **smb.conf(5)** man page

1.9. SETTING UP A SHARE THAT USES WINDOWS ACLS

Samba supports setting Windows ACLs on shares and file system object. This enables you to:

- Use the fine-granular Windows ACLs
- Manage share permissions and file system ACLs using Windows

Alternatively, you can configure a share to use POSIX ACLs.

For details, see [Setting up a Samba file share that uses POSIX ACLs](#) .

Parts of this section were adopted from the [Setting up a Share Using Windows ACLs](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

1.9.1. Granting the SeDiskOperatorPrivilege privilege

Only users and groups having the **SeDiskOperatorPrivilege** privilege granted can configure permissions on shares that use Windows ACLs.

Procedure

1. For example, to grant the **SeDiskOperatorPrivilege** privilege to the **DOMAINDomain Admins** group:

```
# net rpc rights grant "DOMAINDomain Admins" SeDiskOperatorPrivilege -U
"DOMAINadministrator"
Enter DOMAINadministrator's password:
Successfully granted rights.
```




NOTE

In a domain environment, grant **SeDiskOperatorPrivilege** to a domain group. This enables you to centrally manage the privilege by updating a user's group membership.

- To list all users and groups having **SeDiskOperatorPrivilege** granted:

```
# net rpc rights list privileges SeDiskOperatorPrivilege -U "DOMAINadministrator"
Enter administrator's password:
SeDiskOperatorPrivilege:
  BUILTIN\Administrators
  DOMAIN\Domain Admins
```

1.9.2. Enabling Windows ACL support

To configure shares that support Windows ACLs, you must enable this feature in Samba.

Prerequisites

- A user share is configured on the Samba server.

Procedure

- To enable it globally for all shares, add the following settings to the **[global]** section of the **/etc/samba/smb.conf** file:

```
vfs objects = acl_xattr
map acl inherit = yes
store dos attributes = yes
```

Alternatively, you can enable Windows ACL support for individual shares, by adding the same parameters to a share's section instead.

- Restart the **smb** service:

```
# systemctl restart smb
```

1.9.3. Adding a share that uses Windows ACLs

You can create a share named **example**, that shares the content of the **/srv/samba/example/** directory, and uses Windows ACLs.

Procedure

- Create the folder if it does not exist. For example:

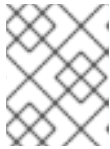
```
# mkdir -p /srv/samba/example/
```

- If you run SELinux in **enforcing** mode, set the **samba_share_t** context on the directory:

```
# semanage fcontext -a -t samba_share_t "/srv/samba/example(/.*)?"
# restorecon -Rv /srv/samba/example/
```

3. Add the example share to the **/etc/samba/smb.conf** file. For example, to add the share write-enabled:

```
[example]
path = /srv/samba/example/
read only = no
```



NOTE

Regardless of the file system ACLs; if you do not set **read only = no**, Samba shares the directory in read-only mode.

4. If you have not enabled Windows ACL support in the **[global]** section for all shares, add the following parameters to the **[example]** section to enable this feature for this share:

```
vfs objects = acl_xattr
map acl inherit = yes
store dos attributes = yes
```

5. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

6. Open the required ports and reload the firewall configuration using the **firewall-cmd** utility:

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

7. Restart the **smb** service:

```
# systemctl restart smb
```

1.9.4. Managing share permissions and file system ACLs of a share that uses Windows ACLs

To manage share permissions and file system ACLs on a Samba share that uses Windows ACLs, use a Windows applications, such as **Computer Management**. For details, see the Windows documentation. Alternatively, use the **smbcacs** utility to manage ACLs.



NOTE

To modify the file system permissions from Windows, you must use an account that has the **SeDiskOperatorPrivilege** privilege granted.

Additional resources

- [Managing ACLs on an SMB share using smbcacs](#)
- [Granting the SeDiskOperatorPrivilege privilege](#)

1.10. MANAGING ACLS ON AN SMB SHARE USING SMBCACLS

The **smbcacs** utility can list, set, and delete ACLs of files and directories stored on an SMB share. You can use **smbcacs** to manage file system ACLs:

- On a local or remote Samba server that uses advanced Windows ACLs or POSIX ACLs
- On Red Hat Enterprise Linux to remotely manage ACLs on a share hosted on Windows

1.10.1. Access control entries

Each ACL entry of a file system object contains Access Control Entries (ACE) in the following format:

```
security_principal:access_right/inheritance_information/permissions
```

Example 1.3. Access control entries

If the **AD\Domain Users** group has **Modify** permissions that apply to **This folder, subfolders, and files** on Windows, the ACL contains the following ACE:

```
AD\Domain Users:ALLOWED/OI|CI/CHANGE
```

An ACE contains the following parts:

Security principal

The security principal is the user, group, or SID the permissions in the ACL are applied to.

Access right

Defines if access to an object is granted or denied. The value can be **ALLOWED** or **DENIED**.

Inheritance information

The following values exist:

Table 1.1. Inheritance settings

Value	Description	Maps to
OI	Object Inherit	This folder and files
CI	Container Inherit	This folder and subfolders
IO	Inherit Only	The ACE does not apply to the current file or directory
ID	Inherited	The ACE was inherited from the parent directory

Additionally, the values can be combined as follows:

Table 1.2. Inheritance settings combinations

Value combinations	Maps to the Windows Applies to setting
OI CI	This folder, subfolders, and files
OI CI IO	Subfolders and files only
CI IO	Subfolders only
OI IO	Files only

Permissions

This value can be either a hex value that represents one or more Windows permissions or an **smbcacls** alias:

- A hex value that represents one or more Windows permissions.
The following table displays the advanced Windows permissions and their corresponding value in hex format:

Table 1.3. Windows permissions and their corresponding smbcacls value in hex format

Windows permissions	Hex values
Full control	0x001F01FF
Traverse folder / execute file	0x00100020
List folder / read data	0x00100001
Read attributes	0x00100080
Read extended attributes	0x00100008
Create files / write data	0x00100002
Create folders / append data	0x00100004
Write attributes	0x00100100
Write extended attributes	0x00100010
Delete subfolders and files	0x00100040
Delete	0x00110000
Read permissions	0x00120000
Change permissions	0x00140000

Windows permissions	Hex values
Take ownership	0x00180000

Multiple permissions can be combined as a single hex value using the bit-wise **OR** operation.

For details, see [ACE mask calculation](#).

- An **smbcacs** alias. The following table displays the available aliases:

Table 1.4. Existing smbcacs aliases and their corresponding Windows permission

smbcacs alias	Maps to Windows permission
R	Read
READ	Read & execute
W	Special: <ul style="list-style-type: none"> ◦ Create files / write data ◦ Create folders / append data ◦ Write attributes ◦ Write extended attributes ◦ Read permissions
D	Delete
P	Change permissions
O	Take ownership
X	Traverse / execute
CHANGE	Modify
FULL	Full control



NOTE

You can combine single-letter aliases when you set permissions. For example, you can set **RD** to apply the Windows permission **Read** and **Delete**. However, you can neither combine multiple non-single-letter aliases nor combine aliases and hex values.

1.10.2. Displaying ACLs using `smbcacs`

To display ACLs on an SMB share, use the **smbcacs** utility. If you run **smbcacs** without any operation parameter, such as **--add**, the utility displays the ACLs of a file system object.

Procedure

For example, to list the ACLs of the root directory of the `//server/example` share:

```
# smbcacs //server/example -U "DOMAINadministrator"
Enter DOMAINadministrator's password:
REVISION:1
CONTROL:SR|PD|DI|DP
OWNER:AD\Administrators
GROUP:AD\Domain Users
ACL:AD\Administrator:ALLOWED/OI|CI/FULL
ACL:AD\Domain Users:ALLOWED/OI|CI/CHANGE
ACL:AD\Domain Guests:ALLOWED/OI|CI/0x00100021
```

The output of the command displays:

- **REVISION:** The internal Windows NT ACL revision of the security descriptor
- **CONTROL:** Security descriptor control
- **OWNER:** Name or SID of the security descriptor's owner
- **GROUP:** Name or SID of the security descriptor's group
- **ACL** entries. For details, see [Access control entries](#).

1.10.3. ACE mask calculation

In most situations, when you add or update an ACE, you use the **smbcacs** aliases listed in [Existing smbcacs aliases and their corresponding Windows permission](#).

However, if you want to set advanced Windows permissions as listed in [Windows permissions and their corresponding smbcacs value in hex format](#), you must use the bit-wise **OR** operation to calculate the correct value. You can use the following shell command to calculate the value:

```
# echo $(printf '0x%X' $(( hex_value_1 | hex_value_2 | ... )))
```

Example 1.4. Calculating an ACE Mask

You want to set the following permissions:

- Traverse folder / execute file (0x00100020)
- List folder / read data (0x00100001)
- Read attributes (0x00100080)

To calculate the hex value for the previous permissions, enter:

```
# echo $(printf '0x%X' $(( 0x00100020 | 0x00100001 | 0x00100080 )))
0x1000A1
```

Use the returned value when you set or update an ACE.

1.10.4. Adding, updating, and removing an ACL using `smbcacs`

Depending on the parameter you pass to the **smbcacs** utility, you can add, update, and remove ACLs from a file or directory.

Adding an ACL

To add an ACL to the root of the `//server/example` share that grants **CHANGE** permissions for **This folder, subfolders, and files** to the **AD\Domain Users** group:

```
# smbcacs //server/example / -U "DOMAIN\administrator --add ACL:"AD\Domain
Users":ALLOWED/OI|CI/CHANGE
```

Updating an ACL

Updating an ACL is similar to adding a new ACL. You update an ACL by overriding the ACL using the **--modify** parameter with an existing security principal. If **smbcacs** finds the security principal in the ACL list, the utility updates the permissions. Otherwise the command fails with an error:

```
ACL for SID principal_name not found
```

For example, to update the permissions of the **AD\Domain Users** group and set them to **READ** for **This folder, subfolders, and files**:

```
# smbcacs //server/example / -U "DOMAIN\administrator --modify ACL:"AD\Domain
Users":ALLOWED/OI|CI/READ
```

Deleting an ACL

To delete an ACL, pass the **--delete** parameter with the exact ACL to the **smbcacs** utility. For example:

```
# smbcacs //server/example / -U "DOMAIN\administrator --delete ACL:"AD\Domain
Users":ALLOWED/OI|CI/READ
```

1.11. ENABLING USERS TO SHARE DIRECTORIES ON A SAMBA SERVER

On a Samba server, you can configure that users can share directories without root permissions.

1.11.1. Enabling the user shares feature

Before users can share directories, the administrator must enable user shares in Samba.

For example, to enable only members of the local **example** group to create user shares.

Procedure

1. Create the local **example** group, if it does not exist:

```
# groupadd example
```

2. Prepare the directory for Samba to store the user share definitions and set its permissions properly. For example:

- a. Create the directory:

```
# mkdir -p /var/lib/samba/usershares/
```

- b. Set write permissions for the **example** group:

```
# chgrp example /var/lib/samba/usershares/  
# chmod 1770 /var/lib/samba/usershares/
```

- c. Set the sticky bit to prevent users to rename or delete files stored by other users in this directory.

3. Edit the **/etc/samba/smb.conf** file and add the following to the **[global]** section:

- a. Set the path to the directory you configured to store the user share definitions. For example:

```
usershare path = /var/lib/samba/usershares/
```

- b. Set how many user shares Samba allows to be created on this server. For example:

```
usershare max shares = 100
```

If you use the default of **0** for the **usershare max shares** parameter, user shares are disabled.

- c. Optionally, set a list of absolute directory paths. For example, to configure that Samba only allows to share subdirectories of the **/data** and **/srv** directory to be shared, set:

```
usershare prefix allow list = /data /srv
```

For a list of further user share-related parameters you can set, see the **USERSHARES** section in the **smb.conf(5)** man page.

4. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

5. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

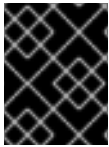
Users are now able to create user shares.

1.11.2. Adding a user share

After you enabled the user share feature in Samba, users can share directories on the Samba server without **root** permissions by running the **net usershare add** command.

Synopsis of the **net usershare add** command:


```
net usershare add share_name path [[ comment ] ] [ ACLs ] [ guest_ok=y|n ]
```



IMPORTANT

If you set ACLs when you create a user share, you must specify the comment parameter prior to the ACLs. To set an empty comment, use an empty string in double quotes.

Note that users can only enable guest access on a user share, if the administrator set **usershare allow guests = yes** in the **[global]** section in the **/etc/samba/smb.conf** file.

Example 1.5. Adding a user share

A user wants to share the **/srv/samba/** directory on a Samba server. The share should be named **example**, have no comment set, and should be accessible by guest users. Additionally, the share permissions should be set to full access for the **AD\Domain Users** group and read permissions for other users. To add this share, run as the user:

```
$ net usershare add example /srv/samba/ "" "AD\Domain Users":F,Everyone:R  
guest_ok=yes
```

1.11.3. Updating settings of a user share

To update settings of a user share, override the share by using the **net usershare add** command with the same share name and the new settings.

See [Adding a user share](#).

1.11.4. Displaying information about existing user shares

Users can enter the **net usershare info** command on a Samba server to display user shares and their settings.

Prerequisites

- A user share is configured on the Samba server.

Procedure

1. To display all user shares created by any user:

```
$ net usershare info -l  
[share_1]  
path=/srv/samba/  
comment=  
usershare_acl=Everyone:R,host_name\user:F,  
guest_ok=y  
...
```

To list only shares created by the user who runs the command, omit the **-l** parameter.

2. To display only the information about specific shares, pass the share name or wild cards to the command. For example, to display the information about shares whose name starts with **share_**:

```
$ net usershare info -l share_*
```

1.11.5. Listing user shares

If you want to list only the available user shares without their settings on a Samba server, use the **net usershare list** command.

Prerequisites

- A user share is configured on the Samba server.

Procedure

1. To list the shares created by any user:

```
$ net usershare list -l  
share_1  
share_2  
...
```

To list only shares created by the user who runs the command, omit the **-l** parameter.

2. To list only specific shares, pass the share name or wild cards to the command. For example, to list only shares whose name starts with **share_**:

```
$ net usershare list -l share_*
```

1.11.6. Deleting a user share

To delete a user share, use the command **net usershare delete** command as the user who created the share or as the **root** user.

Prerequisites

- A user share is configured on the Samba server.

Procedure

```
$ net usershare delete share_name
```

1.12. CONFIGURING A SHARE TO ALLOW ACCESS WITHOUT AUTHENTICATION

In certain situations, you want to share a directory to which users can connect without authentication. To configure this, enable guest access on a share.

**WARNING**

Shares that do not require authentication can be a security risk.

1.12.1. Enabling guest access to a share

If guest access is enabled on a share, Samba maps guest connections to the operating system account set in the **guest account** parameter. Guest users can access files on this share if at least one of the following conditions is satisfied:

- The account is listed in file system ACLs
- The POSIX permissions for **other** users allow it

Example 1.6. Guest share permissions

If you configured Samba to map the guest account to **nobody**, which is the default, the ACLs in the following example:

- Allow guest users to read **file1.txt**
- Allow guest users to read and modify **file2.txt**
- Prevent guest users to read or modify **file3.txt**

```
-rw-r--r--. 1 root    root    1024 1. Sep 10:00 file1.txt
-rw-r-----. 1 nobody root    1024 1. Sep 10:00 file2.txt
-rw-r-----. 1 root    root    1024 1. Sep 10:00 file3.txt
```

Procedure

1. Edit the **/etc/samba/smb.conf** file:
 - a. If this is the first guest share you set up on this server:
 - i. Set **map to guest = Bad User** in the **[global]** section:

```
[global]
...
map to guest = Bad User
```

With this setting, Samba rejects login attempts that use an incorrect password unless the user name does not exist. If the specified user name does not exist and guest access is enabled on a share, Samba treats the connection as a guest log in.

- ii. By default, Samba maps the guest account to the **nobody** account on Red Hat Enterprise Linux. Alternatively, you can set a different account. For example:

```
[global]
...
guest account = user_name
```

The account set in this parameter must exist locally on the Samba server. For security reasons, Red Hat recommends using an account that does not have a valid shell assigned.

- b. Add the **guest ok = yes** setting to the **[example]** share section:

```
[example]
...
guest ok = yes
```

2. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

3. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

1.13. CONFIGURING SAMBA FOR MACOS CLIENTS

The **fruit** virtual file system (VFS) Samba module provides enhanced compatibility with Apple server message block (SMB) clients.

1.13.1. Optimizing the Samba configuration for providing file shares for macOS clients

This section describes how to configure the **fruit** module for all Samba shares hosted on the server to optimize Samba file shares for macOS clients.



NOTE

Red Hat recommends enabling the **fruit** module globally. Clients using macOS negotiate the server server message block version 2 (SMB2) Apple (AAPL) protocol extensions when the client establishes the first connection to the server. If the client first connects to a share without AAPL extensions enabled, the client does not use the extensions for any share of the server.

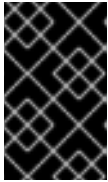
Prerequisites

- Samba is configured as a file server.

Procedure

1. Edit the **/etc/samba/smb.conf** file, and enable the **fruit** and **streams_xattr** VFS modules in the **[global]** section:

```
vfs objects = fruit streams_xattr
```



IMPORTANT

You must enable the **fruit** module before enabling **streams_xattr**. The **fruit** module uses alternate data streams (ADS). For this reason, you must also enable the **streams_xattr** module.

- Optionally, to provide macOS Time Machine support on a share, add the following setting to the share configuration in the **/etc/samba/smb.conf** file:

```
fruit:time machine = yes
```

- Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

- Reload the Samba configuration:

```
# smbcontrol all reload-config
```

Additional resources

- **vfs_fruit(8)** man page.
- Configuring file shares:
 - [Setting up a Samba file share that uses POSIX ACLs](#)
 - [Setting up a share that uses Windows ACLs](#) .

1.14. USING THE SMBCLIENT UTILITY TO ACCESS AN SMB SHARE

The **smbclient** utility enables you to access file shares on an SMB server, similarly to a command-line FTP client. You can use it, for example, to upload and download files to and from a share.

Prerequisites

- The **samba-client** package is installed.

1.14.1. How the smbclient interactive mode works

For example, to authenticate to the **example** share hosted on **server** using the **DOMAIN\user** account:

```
# smbclient -U "DOMAIN\user" //server/example
Enter domain\user's password:
Try "help" to get a list of possible commands.
smb: \>
```

After **smbclient** connected successfully to the share, the utility enters the interactive mode and shows the following prompt:

```
smb: \>
```

To display all available commands in the interactive shell, enter:

```
smb: \> help
```

To display the help for a specific command, enter:

```
smb: \> help command_name
```

Additional resources

- **smbclient(1)** man page

1.14.2. Using smbclient in interactive mode

If you use **smbclient** without the **-c** parameter, the utility enters the interactive mode. The following procedure shows how to connect to an SMB share and download a file from a subdirectory.

Procedure

1. Connect to the share:

```
# smbclient -U "DOMAIN\user_name" //server_name/share_name
```

2. Change into the **/example/** directory:

```
smb: \> d /example/
```

3. List the files in the directory:

```
smb: \example\> ls
.          D      0 Thu Nov 1 10:00:00 2018
..         D      0 Thu Nov 1 10:00:00 2018
example.txt N 1048576 Thu Nov 1 10:00:00 2018

9950208 blocks of size 1024. 8247144 blocks available
```

4. Download the **example.txt** file:

```
smb: \example\> get example.txt
getting file \directory\subdirectory\example.txt of size 1048576 as example.txt (511975,0
KiloBytes/sec) (average 170666,7 KiloBytes/sec)
```

5. Disconnect from the share:

```
smb: \example\> exit
```

1.14.3. Using smbclient in scripting mode

If you pass the **-c** parameter to **smbclient**, you can automatically execute the commands on the remote SMB share. This enables you to use **smbclient** in scripts.

The following procedure shows how to connect to an SMB share and download a file from a subdirectory.

Procedure

- Use the following command to connect to the share, change into the **example** directory, download the **example.txt** file:

```
# smbclient -U DOMAIN\user_name //server_name/share_name -c "cd /example/ ; get example.txt ; exit"
```

1.15. SETTING UP SAMBA AS A PRINT SERVER

If you set up Samba as a print server, clients in your network can use Samba to print. Additionally, Windows clients can, if configured, download the driver from the Samba server.

Parts of this section were adopted from the [Setting up Samba as a Print Server](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

Prerequisites

Samba has been set up in one of the following modes:

- [Standalone server](#)
- [Domain member](#)

1.15.1. Enabling print server support in Samba

By default, print server support is not enabled in Samba. To use Samba as a print server, you must configure Samba accordingly.



NOTE

Print jobs and printer operations require remote procedure calls (RPCs). By default, Samba starts the **rpcd_spoolss** service on demand to manage RPCs. During the first RPC call, or when you update the printer list in CUPS, Samba retrieves the printer information from CUPS. This can require approximately 1 second per printer. Therefore, if you have more than 50 printers, tune the **rpcd_spoolss** settings.

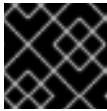
Prerequisites

- The printers are configured in a CUPS server.
For details about configuring printers in CUPS, see the documentation provided in the CUPS web console (<https://printserver:631/help>) on the print server.

Procedure

1. Edit the **/etc/samba/smb.conf** file:
 - a. Add the **[printers]** section to enable the printing backend in Samba:

```
[printers]
comment = All Printers
path = /var/tmp/
printable = yes
create mask = 0600
```

**IMPORTANT**

The **[printers]** share name is hard-coded and cannot be changed.

- b. If the CUPS server runs on a different host or port, specify the setting in the **[printers]** section:

```
cups server = printserver.example.com:631
```

- c. If you have many printers, set the number of idle seconds to a higher value than the numbers of printers connected to CUPS. For example, if you have 100 printers, set in the **[global]** section:

```
rpcd_spoolss:idle_seconds = 200
```

If this setting does not scale in your environment, also increase the number of **rpcd_spoolss** workers in the **[global]** section:

```
rpcd_spoolss:num_workers = 10
```

By default, **rpcd_spoolss** starts 5 workers.

2. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

3. Open the required ports and reload the firewall configuration using the **firewall-cmd** utility:

```
# firewall-cmd --permanent --add-service=samba  
# firewall-cmd --reload
```

4. Restart the **smb** service:

```
# systemctl restart smb
```

After restarting the service, Samba automatically shares all printers that are configured in the CUPS back end. If you want to manually share only specific printers, see [Manually sharing specific printers](#).

Verification

- Submit a print job. For example, to print a PDF file, enter:

```
# smbclient -Uuser //sambaserver.example.com/printer_name -c "print example.pdf"
```

1.15.2. Manually sharing specific printers

If you configured Samba as a print server, by default, Samba shares all printers that are configured in the CUPS back end. The following procedure explains how to share only specific printers.

Prerequisites

- Samba is set up as a print server

Procedure

1. Edit the `/etc/samba/smb.conf` file:

- a. In the **[global]** section, disable automatic printer sharing by setting:

```
load printers = no
```

- b. Add a section for each printer you want to share. For example, to share the printer named **example** in the CUPS back end as **Example-Printer** in Samba, add the following section:

```
[Example-Printer]
  path = /var/tmp/
  printable = yes
  printer name = example
```

You do not need individual spool directories for each printer. You can set the same spool directory in the **path** parameter for the printer as you set in the **[printers]** section.

2. Verify the `/etc/samba/smb.conf` file:

```
# testparm
```

3. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

1.16. SETTING UP AUTOMATIC PRINTER DRIVER DOWNLOADS FOR WINDOWS CLIENTS ON SAMBA PRINT SERVERS

If you are running a Samba print server for Windows clients, you can upload drivers and preconfigure printers. If a user connects to a printer, Windows automatically downloads and installs the driver locally on the client. The user does not require local administrator permissions for the installation. Additionally, Windows applies preconfigured driver settings, such as the number of trays.

Parts of this section were adopted from the [Setting up Automatic Printer Driver Downloads for Windows Clients](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

Prerequisites

- Samba is set up as a print server

1.16.1. Basic information about printer drivers

This section provides general information about printer drivers.

Supported driver model version

Samba only supports the printer driver model version 3 which is supported in Windows 2000 and later,

and Windows Server 2000 and later. Samba does not support the driver model version 4, introduced in Windows 8 and Windows Server 2012. However, these and later Windows versions also support version 3 drivers.

Package-aware drivers

Samba does not support package-aware drivers.

Preparing a printer driver for being uploaded

Before you can upload a driver to a Samba print server:

- Unpack the driver if it is provided in a compressed format.
- Some drivers require to start a setup application that installs the driver locally on a Windows host. In certain situations, the installer extracts the individual files into the operating system's temporary folder during the setup runs. To use the driver files for uploading:
 - a. Start the installer.
 - b. Copy the files from the temporary folder to a new location.
 - c. Cancel the installation.

Ask your printer manufacturer for drivers that support uploading to a print server.

Providing 32-bit and 64-bit drivers for a printer to a client

To provide the driver for a printer for both 32-bit and 64-bit Windows clients, you must upload a driver with exactly the same name for both architectures. For example, if you are uploading the 32-bit driver named **Example PostScript** and the 64-bit driver named **Example PostScript (v1.0)**, the names do not match. Consequently, you can only assign one of the drivers to a printer and the driver will not be available for both architectures.

1.16.2. Enabling users to upload and preconfigure drivers

To be able to upload and preconfigure printer drivers, a user or a group needs to have the **SePrintOperatorPrivilege** privilege granted. A user must be added into the **printadmin** group. Red Hat Enterprise Linux automatically creates this group when you install the **samba** package. The **printadmin** group gets assigned the lowest available dynamic system GID that is lower than 1000.

Procedure

1. For example, to grant the **SePrintOperatorPrivilege** privilege to the **printadmin** group:

```
# net rpc rights grant "printadmin" SePrintOperatorPrivilege -U
"DOMAINadministrator"
Enter DOMAINadministrator's password:
Successfully granted rights.
```



NOTE

In a domain environment, grant **SePrintOperatorPrivilege** to a domain group. This enables you to centrally manage the privilege by updating a user's group membership.

2. To list all users and groups having **SePrintOperatorPrivilege** granted:

```
# net rpc rights list privileges SePrintOperatorPrivilege -U "DOMAINadministrator"
Enter administrator's password:
SePrintOperatorPrivilege:
    BUILTIN\Administrators
    DOMAIN\printadmin
```

1.16.3. Setting up the print\$ share

Windows operating systems download printer drivers from a share named **print\$** from a print server. This share name is hard-coded in Windows and cannot be changed.

The following procedure explains how to share the **/var/lib/samba/drivers/** directory as **print\$**, and enable members of the local **printadmin** group to upload printer drivers.

Procedure

1. Add the **[print\$]** section to the **/etc/samba/smb.conf** file:

```
[print$]
    path = /var/lib/samba/drivers/
    read only = no
    write list = @printadmin
    force group = @printadmin
    create mask = 0664
    directory mask = 2775
```

Using these settings:

- Only members of the **printadmin** group can upload printer drivers to the share.
 - The group of new created files and directories will be set to **printadmin**.
 - The permissions of new files will be set to **664**.
 - The permissions of new directories will be set to **2775**.
2. To upload only 64-bit drivers for all printers, include this setting in the **[global]** section in the **/etc/samba/smb.conf** file:

```
spoolss: architecture = Windows x64
```

Without this setting, Windows only displays drivers for which you have uploaded at least the 32-bit version.

3. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

4. Reload the Samba configuration

```
# smbcontrol all reload-config
```

5. Create the **printadmin** group if it does not exist:

```
# groupadd printadmin
```

- Grant the **SePrintOperatorPrivilege** privilege to the **printadmin** group.

```
# net rpc rights grant "printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
```

Enter *DOMAIN\administrator's* password:
Successfully granted rights.

- If you run SELinux in **enforcing** mode, set the **samba_share_t** context on the directory:

```
# semanage fcontext -a -t samba_share_t "/var/lib/samba/drivers(/.)*" # *restorecon -
Rv /var/lib/samba/drivers/
```

- Set the permissions on the **/var/lib/samba/drivers/** directory:

- If you use POSIX ACLs, set:

```
# chgrp -R "printadmin" /var/lib/samba/drivers/
# chmod -R 2775 /var/lib/samba/drivers/
```

- If you use Windows ACLs, set:

Principal	Access	Apply to
CREATOR OWNER	Full control	Subfolders and files only
Authenticated Users	Read & execute, List folder contents, Read	This folder, subfolders, and files
printadmin	Full control	This folder, subfolders, and files

For details about setting ACLs on Windows, see the Windows documentation.

Additional resources

- [Enabling users to upload and preconfigure drivers.](#)

1.16.4. Creating a GPO to enable clients to trust the Samba print server

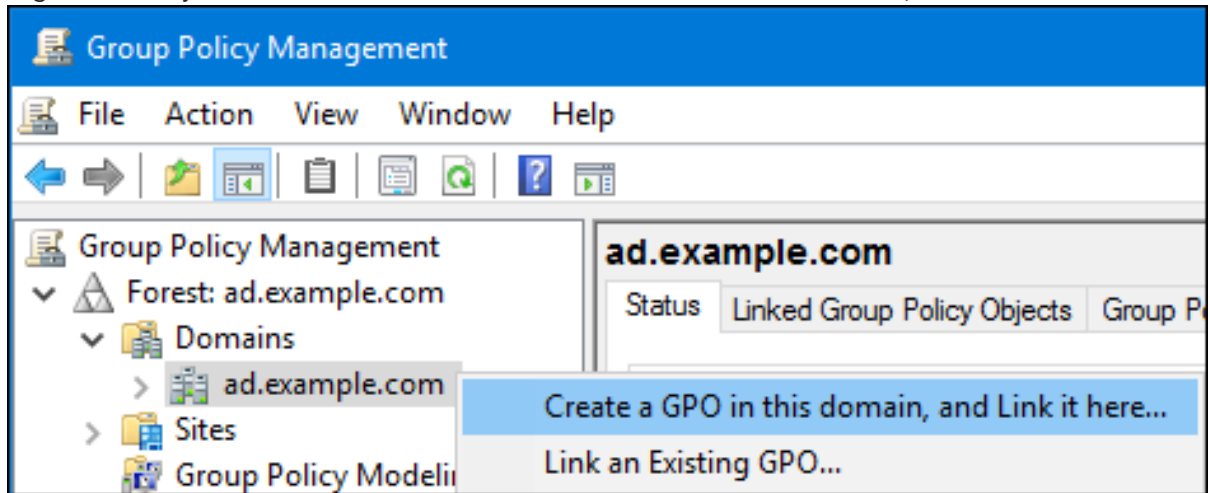
For security reasons, recent Windows operating systems prevent clients from downloading non-package-aware printer drivers from an untrusted server. If your print server is a member in an AD, you can create a Group Policy Object (GPO) in your domain to trust the Samba server.

Prerequisites

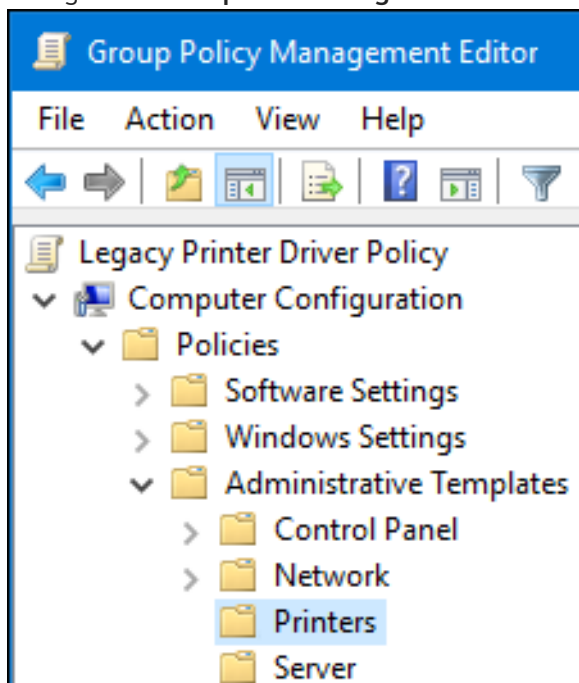
- The Samba print server is a member of an AD domain.
- The Windows computer you are using to create the GPO must have the Windows Remote Server Administration Tools (RSAT) installed. For details, see the Windows documentation.

Procedure

1. Log into a Windows computer using an account that is allowed to edit group policies, such as the AD domain **Administrator** user.
2. Open the **Group Policy Management Console**.
3. Right-click to your AD domain and select **Create a GPO in this domain, and Link it here**.



4. Enter a name for the GPO, such as **Legacy Printer Driver Policy** and click **OK**. The new GPO will be displayed under the domain entry.
5. Right-click to the newly-created GPO and select **Edit** to open the **Group Policy Management Editor**.
6. Navigate to **Computer Configuration → Policies → Administrative Templates → Printers**.



7. On the right side of the window, double-click **Point and Print Restriction** to edit the policy:
 - a. Enable the policy and set the following options:
 - i. Select **Users can only point and print to these servers** and enter the fully-qualified domain name (FQDN) of the Samba print server to the field next to this option.

- ii. In both check boxes under **Security Prompts**, select **Do not show warning or elevation prompt**.

Point and Print Restrictions

Point and Print Restrictions

☐ Not Configured Comment:
☒ Enabled
☐ Disabled

Supported on: **At least Windows Vista**

Options:

☒ Users can only point and print to these servers:
 Enter fully qualified server names separated by semicolons

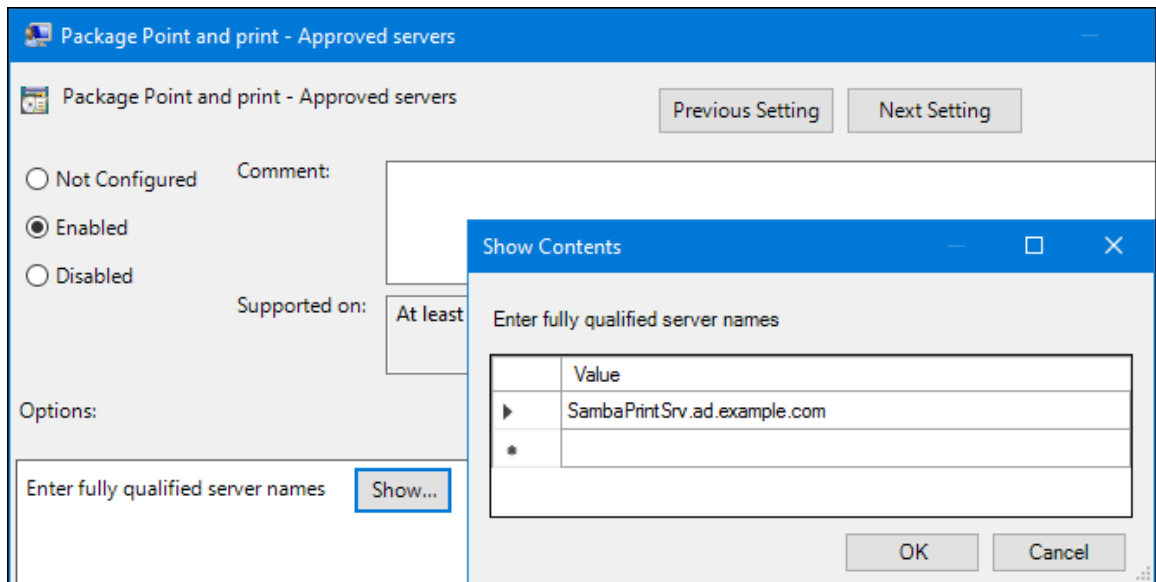
☐ Users can only point and print to machines in their forest

Security Prompts:

When installing drivers for a new connection:

When updating drivers for an existing connection:

- b. Click OK.
8. Double-click **Package Point and Print - Approved servers** to edit the policy:
 - a. Enable the policy and click the **Show** button.
 - b. Enter the FQDN of the Samba print server.



c. Close both the **Show Contents** and the policy's properties window by clicking **OK**.

9. Close the **Group Policy Management Editor**.

10. Close the **Group Policy Management Console**.

After the Windows domain members applied the group policy, printer drivers are automatically downloaded from the Samba server when a user connects to a printer.

Additional resources

- For using group policies, see the Windows documentation.

1.16.5. Uploading drivers and preconfiguring printers

Use the **Print Management** application on a Windows client to upload drivers and preconfigure printers hosted on the Samba print server. For further details, see the Windows documentation.

1.17. RUNNING SAMBA ON A SERVER WITH FIPS MODE ENABLED

This section provides an overview of the limitations of running Samba with FIPS mode enabled. It also provides the procedure for enabling FIPS mode on a Red Hat Enterprise Linux host running Samba.

1.17.1. Limitations of using Samba in FIPS mode

The following Samba modes and features work in FIPS mode under the indicated conditions:

- Samba as a domain member only in Active Directory (AD) or Red Hat Identity Management (IdM) environments with Kerberos authentication that uses AES ciphers.
- Samba as a file server on an Active Directory domain member. However, this requires that clients use Kerberos to authenticate to the server.

Due to the increased security of FIPS, the following Samba features and modes do not work if FIPS mode is enabled:

- NT LAN Manager (NTLM) authentication because RC4 ciphers are blocked

- The server message block version 1 (SMB1) protocol
- The stand-alone file server mode because it uses NTLM authentication
- NT4-style domain controllers
- NT4-style domain members. Note that Red Hat continues supporting the primary domain controller (PDC) functionality IdM uses in the background.
- Password changes against the Samba server. You can only perform password changes using Kerberos against an Active Directory domain controller.

The following feature is not tested in FIPS mode and, therefore, is not supported by Red Hat:

- Running Samba as a print server

1.17.2. Using Samba in FIPS mode

You can enable the FIPS mode on a RHEL host that runs Samba.

Prerequisites

- Samba is configured on the Red Hat Enterprise Linux host.
- Samba runs in a mode that is supported in FIPS mode.

Procedure

1. Enable the FIPS mode on RHEL:

```
# fips-mode-setup --enable
```

2. Reboot the server:

```
# reboot
```

3. Use the **testparm** utility to verify the configuration:

```
# testparm -s
```

If the command displays any errors or incompatibilities, fix them to ensure that Samba works correctly.

Additional resources

- [Section 1.17.1, “Limitations of using Samba in FIPS mode”](#)

1.18. TUNING THE PERFORMANCE OF A SAMBA SERVER

Learn what settings can improve the performance of Samba in certain situations, and which settings can have a negative performance impact.

Parts of this section were adopted from the [Performance Tuning](#) documentation published in the Samba Wiki. License: [CC BY 4.0](#). Authors and contributors: See the [history](#) tab on the Wiki page.

Prerequisites

- Samba is set up as a file or print server

1.18.1. Setting the SMB protocol version

Each new SMB version adds features and improves the performance of the protocol. The recent Windows and Windows Server operating systems always supports the latest protocol version. If Samba also uses the latest protocol version, Windows clients connecting to Samba benefit from the performance improvements. In Samba, the default value of the server max protocol is set to the latest supported stable SMB protocol version.



NOTE

To always have the latest stable SMB protocol version enabled, do not set the **server max protocol** parameter. If you set the parameter manually, you will need to modify the setting with each new version of the SMB protocol, to have the latest protocol version enabled.

The following procedure explains how to use the default value in the **server max protocol** parameter.

Procedure

1. Remove the **server max protocol** parameter from the **[global]** section in the **/etc/samba/smb.conf** file.
2. Reload the Samba configuration

```
# smbcontrol all reload-config
```

1.18.2. Tuning shares with directories that contain a large number of files

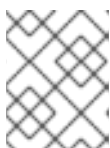
Linux supports case-sensitive file names. For this reason, Samba needs to scan directories for uppercase and lowercase file names when searching or accessing a file. You can configure a share to create new files only in lowercase or uppercase, which improves the performance.

Prerequisites

- Samba is configured as a file server

Procedure

1. Rename all files on the share to lowercase.



NOTE

Using the settings in this procedure, files with names other than in lowercase will no longer be displayed.

2. Set the following parameters in the share's section:

```
case sensitive = true
default case = lower
```

```
preserve case = no
short preserve case = no
```

For details about the parameters, see their descriptions in the **smb.conf(5)** man page.

3. Verify the **/etc/samba/smb.conf** file:

```
# testparm
```

4. Reload the Samba configuration:

```
# smbcontrol all reload-config
```

After you applied these settings, the names of all newly created files on this share use lowercase. Because of these settings, Samba no longer needs to scan the directory for uppercase and lowercase, which improves the performance.

1.18.3. Settings that can have a negative performance impact

By default, the kernel in Red Hat Enterprise Linux is tuned for high network performance. For example, the kernel uses an auto-tuning mechanism for buffer sizes. Setting the **socket options** parameter in the **/etc/samba/smb.conf** file overrides these kernel settings. As a result, setting this parameter decreases the Samba network performance in most cases.

To use the optimized settings from the Kernel, remove the **socket options** parameter from the **[global]** section in the **/etc/samba/smb.conf**.

1.19. CONFIGURING SAMBA TO BE COMPATIBLE WITH CLIENTS THAT REQUIRE AN SMB VERSION LOWER THAN THE DEFAULT

Samba uses a reasonable and secure default value for the minimum server message block (SMB) version it supports. However, if you have clients that require an older SMB version, you can configure Samba to support it.

1.19.1. Setting the minimum SMB protocol version supported by a Samba server

In Samba, the **server min protocol** parameter in the **/etc/samba/smb.conf** file defines the minimum server message block (SMB) protocol version the Samba server supports. You can change the minimum SMB protocol version.



NOTE

By default, Samba on RHEL 8.2 and later supports only SMB2 and newer protocol versions. Red Hat recommends to not use the deprecated SMB1 protocol. However, if your environment requires SMB1, you can manually set the **server min protocol** parameter to **NT1** to re-enable SMB1.

Prerequisites

- Samba is installed and configured.

Procedure

1. Edit the **/etc/samba/smb.conf** file, add the **server min protocol** parameter, and set the parameter to the minimum SMB protocol version the server should support. For example, to set the minimum SMB protocol version to **SMB3**, add:

```
server min protocol = SMB3
```

2. Restart the **smb** service:

```
# systemctl restart smb
```

Additional resources

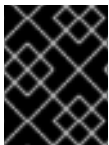
- **smb.conf(5)** man page

1.20. FREQUENTLY USED SAMBA COMMAND-LINE UTILITIES

This chapter describes frequently used commands when working with a Samba server.

1.20.1. Using the **net ads join** and **net rpc join** commands

Using the **join** subcommand of the **net** utility, you can join Samba to an AD or NT4 domain. To join the domain, you must create the **/etc/samba/smb.conf** file manually, and optionally update additional configurations, such as PAM.



IMPORTANT

Red Hat recommends using the **realm** utility to join a domain. The **realm** utility automatically updates all involved configuration files.

Procedure

1. Manually create the **/etc/samba/smb.conf** file with the following settings:

- For an AD domain member:

```
[global]
workgroup = domain_name
security = ads
passdb backend = tdbsam
realm = AD_REALM
```

- For an NT4 domain member:

```
[global]
workgroup = domain_name
security = user
passdb backend = tdbsam
```

2. Add an ID mapping configuration for the ***** default domain and for the domain you want to join to the **[global]** section in the **/etc/samba/smb.conf** file.
3. Verify the **/etc/samba/smb.conf** file:

testparm

4. Join the domain as the domain administrator:

- To join an AD domain:

```
# net ads join -U "DOMAIN\administrator"
```

- To join an NT4 domain:

```
# net rpc join -U "DOMAIN\administrator"
```

5. Append the **winbind** source to the **passwd** and **group** database entry in the **/etc/nsswitch.conf** file:

```
passwd:  files winbind
group:   files winbind
```

6. Enable and start the **winbind** service:

```
# systemctl enable --now winbind
```

7. Optionally, configure PAM using the **authselect** utility.

For details, see the **authselect(8)** man page.

8. Optionally for AD environments, configure the Kerberos client.

For details, see the documentation of your Kerberos client.

Additional resources

- [Joining Samba to a domain](#) .
- [Understanding and configuring Samba ID mapping](#) .

1.20.2. Using the net rpc rights command

In Windows, you can assign privileges to accounts and groups to perform special operations, such as setting ACLs on a share or upload printer drivers. On a Samba server, you can use the **net rpc rights** command to manage privileges.

Listing privileges you can set

To list all available privileges and their owners, use the **net rpc rights list** command. For example:

```
# net rpc rights list -U "DOMAIN\administrator"
```

Enter *DOMAIN\administrator's* password:

```
SeMachineAccountPrivilege  Add machines to domain
SeTakeOwnershipPrivilege   Take ownership of files or other objects
    SeBackupPrivilege       Back up files and directories
    SeRestorePrivilege      Restore files and directories
SeRemoteShutdownPrivilege  Force shutdown from a remote system
SePrintOperatorPrivilege   Manage printers
```

```
SeAddUsersPrivilege  Add users and groups to the domain
SeDiskOperatorPrivilege  Manage disk shares
SeSecurityPrivilege  System security
```

Granting privileges

To grant a privilege to an account or group, use the **net rpc rights grant** command.

For example, grant the **SePrintOperatorPrivilege** privilege to the **DOMAIN\printadmin** group:

```
# net rpc rights grant "DOMAIN\printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully granted rights.
```

Revoking privileges

To revoke a privilege from an account or group, use the **net rpc rights revoke** command.

For example, to revoke the **SePrintOperatorPrivilege** privilege from the **DOMAIN\printadmin** group:

```
# net rpc rights remove "DOMAIN\printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully revoked rights.
```

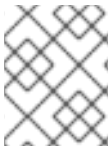
1.20.3. Using the net rpc share command

The **net rpc share** command provides the capability to list, add, and remove shares on a local or remote Samba or Windows server.

Listing shares

To list the shares on an SMB server, use the **net rpc share list** command. Optionally, pass the **-S** *server_name* parameter to the command to list the shares of a remote server. For example:

```
# net rpc share list -U "DOMAIN\administrator" -S server_name
Enter DOMAIN\administrator's password:
IPC$
share_1
share_2
...
```



NOTE

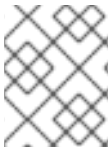
Shares hosted on a Samba server that have **browseable = no** set in their section in the **/etc/samba/smb.conf** file are not displayed in the output.

Adding a share

The **net rpc share add** command enables you to add a share to an SMB server.

For example, to add a share named **example** on a remote Windows server that shares the **C:\example** directory:

```
# net rpc share add example="C:\example" -U "DOMAIN\administrator" -S server_name
```

**NOTE**

You must omit the trailing backslash in the path when specifying a Windows directory name.

To use the command to add a share to a Samba server:

- The user specified in the **-U** parameter must have the **SeDiskOperatorPrivilege** privilege granted on the destination server.
- You must write a script that adds a share section to the **/etc/samba/smb.conf** file and reloads Samba. The script must be set in the **add share command** parameter in the **[global]** section in **/etc/samba/smb.conf**. For further details, see the **add share command** description in the **smb.conf(5)** man page.

Removing a share

The **net rpc share delete** command enables you to remove a share from an SMB server.

For example, to remove the share named example from a remote Windows server:

```
# net rpc share delete example -U "DOMAINadministrator" -S server_name
```

To use the command to remove a share from a Samba server:

- The user specified in the **-U** parameter must have the **SeDiskOperatorPrivilege** privilege granted.
- You must write a script that removes the share's section from the **/etc/samba/smb.conf** file and reloads Samba. The script must be set in the **delete share command** parameter in the **[global]** section in **/etc/samba/smb.conf**. For further details, see the **delete share command** description in the **smb.conf(5)** man page.

1.20.4. Using the net user command

The **net user** command enables you to perform the following actions on an AD DC or NT4 PDC:

- List all user accounts
- Add users
- Remove Users

**NOTE**

Specifying a connection method, such as **ads** for AD domains or **rpc** for NT4 domains, is only required when you list domain user accounts. Other user-related subcommands can auto-detect the connection method.

Pass the **-U user_name** parameter to the command to specify a user that is allowed to perform the requested action.

Listing domain user accounts

To list all users in an AD domain:

```
# net ads user -U "DOMAINadministrator"
```

To list all users in an NT4 domain:

```
# net rpc user -U "DOMAINadministrator"
```

Adding a user account to the domain

On a Samba domain member, you can use the **net user add** command to add a user account to the domain.

For example, add the **user** account to the domain:

1. Add the account:

```
# net user add user password -U "DOMAINadministrator"
User user added
```

2. Optionally, use the remote procedure call (RPC) shell to enable the account on the AD DC or NT4 PDC. For example:

```
# net rpc shell -U DOMAINadministrator -S DC_or_PDC_name
Talking to domain DOMAIN (S-1-5-21-1424831554-512457234-5642315751)

net rpc> user edit disabled user: no
Set user's disabled flag from [yes] to [no]

net rpc> exit
```

Deleting a user account from the domain

On a Samba domain member, you can use the **net user delete** command to remove a user account from the domain.

For example, to remove the **user** account from the domain:

```
# net user delete user -U "DOMAINadministrator"
User user deleted
```

1.20.5. Using the rpcclient utility

The **rpcclient** utility enables you to manually execute client-side Microsoft Remote Procedure Call (MS-RPC) functions on a local or remote SMB server. However, most of the features are integrated into separate utilities provided by Samba. Use **rpcclient** only for testing MS-PRC functions.

Prerequisites

- The **samba-client** package is installed.

Examples

For example, you can use the **rpcclient** utility to:

- Manage the printer Spool Subsystem (SPOOLSS).

Example 1.7. Assigning a Driver to a Printer

```
# rpcclient server_name -U "DOMAINadministrator" -c 'setdriver "printer_name"
"driver_name"
```

```
Enter DOMAINadministrators password:  
Successfully set printer_name to driver driver_name.
```

- Retrieve information about an SMB server.

Example 1.8. Listing all File Shares and Shared Printers

```
# rpcclient server_name -U "DOMAINadministrator" -c 'netshareenum'  
Enter DOMAINadministrators password:  
netname: Example_Share  
remark:  
path: C:\srv\samba\example_share\  
password:  
netname: Example_Printer  
remark:  
path: C:\var\spool\samba\  
password:
```

- Perform actions using the Security Account Manager Remote (SAMR) protocol.

Example 1.9. Listing Users on an SMB Server

```
# rpcclient server_name -U "DOMAINadministrator" -c 'enumdomusers'  
Enter DOMAINadministrators password:  
user:[user1] rid:[0x3e8]  
user:[user2] rid:[0x3e9]
```

If you run the command against a standalone server or a domain member, it lists the users in the local database. Running the command against an AD DC or NT4 PDC lists the domain users.

Additional resources

- **rpcclient(1)** man page

1.20.6. Using the samba-regedit application

Certain settings, such as printer configurations, are stored in the registry on the Samba server. You can use the ncurses-based **samba-regedit** application to edit the registry of a Samba server.

Path: ...AL_MACHINE/SOFTWARE/Microsoft/Windows NT/CurrentVersion/Print/Printers/

Key	Value
Name	Name Type Data
+Example-Printer	Attributes REG_DWORD 0x00001848 (6216)
	ChangeID REG_DWORD 0x00160374 (1442676)
	Datatype REG_SZ RAW
	Default Priority REG_DWORD 0x00000001 (1)
	Description REG_SZ
	Location REG_SZ
	Name REG_SZ Example-Printer
	Parameters REG_SZ
	Port REG_SZ Samba Printer Port
	Print Processor REG_SZ winprint
	Printer Driver REG_SZ Example Printer Driver
	Priority REG_DWORD 0x00000001 (1)
	Security REG_BINARY (248 bytes)
	Separator File REG_SZ
	Share Name REG_SZ Example-Printer
	StartTime REG_DWORD 0x00000000 (0)
	Status REG_DWORD 0x00000000 (0)
	UntilTime REG_DWORD 0x00000000 (0)

[n] New Value [d] Del Value [ENTER] Edit [b] Edit binary **VALUES**
[TAB] Switch sections [q] Quit [UP] List up [DOWN] List down [/] Search [x] Next

Prerequisites

- The **samba-client** package is installed.

Procedure

To start the application, enter:

```
# samba-regedit
```

Use the following keys:

- Cursor up and cursor down: Navigate through the registry tree and the values.
- **Enter**: Opens a key or edits a value.
- **Tab**: Switches between the **Key** and **Value** pane.
- **Ctrl+C**: Closes the application.

1.20.7. Using the smbcontrol utility

The **smbcontrol** utility enables you to send command messages to the **smbd**, **nmbd**, **winbindd**, or all of these services. These control messages instruct the service, for example, to reload its configuration.

Prerequisites

- The **samba-common-tools** package is installed.

Procedure

- Reload the configuration of the **smbd**, **nmbd**, **winbindd** services by sending the **reload-config** message type to the **all** destination:

■

smbcontrol all reload-config

Additional resources

- **smbcontrol(1)** man page

1.20.8. Using the **smbpasswd** utility

The **smbpasswd** utility manages user accounts and passwords in the local Samba database.

Prerequisites

- The **samba-common-tools** package is installed.

Procedure

1. If you run the command as a user, **smbpasswd** changes the Samba password of the user who run the command. For example:

```
[user@server ~]$ smbpasswd
New SMB password: password
Retype new SMB password: password
```

2. If you run **smbpasswd** as the **root** user, you can use the utility, for example, to:

- Create a new user:

```
[root@server ~]# smbpasswd -a user_name
New SMB password: password
Retype new SMB password: password
Added user user_name.
```



NOTE

Before you can add a user to the Samba database, you must create the account in the local operating system. See the [Adding a new user from the command line](#) section in the Configuring basic system settings guide.

- Enable a Samba user:

```
[root@server ~]# smbpasswd -e user_name
Enabled user user_name.
```

- Disable a Samba user:

```
[root@server ~]# smbpasswd -x user_name
Disabled user user_name
```

- Delete a user:

```
[root@server ~]# smbpasswd -x user_name
Deleted user user_name.
```

Additional resources

- **smbpasswd(8)** man page

1.20.9. Using the smbstatus utility

The **smbstatus** utility reports on:

- Connections per PID of each **smbd** daemon to the Samba server. This report includes the user name, primary group, SMB protocol version, encryption, and signing information.
- Connections per Samba share. This report includes the PID of the **smbd** daemon, the IP of the connecting machine, the time stamp when the connection was established, encryption, and signing information.
- A list of locked files. The report entries include further details, such as opportunistic lock (oplock) types

Prerequisites

- The **samba** package is installed.
- The **smbd** service is running.

Procedure

smbstatus

Samba version 4.15.2

PID	Username	Group	Machine	Protocol Version	Encryption	Signing
-----	----------	-------	---------	------------------	------------	---------

.....
-

963	DOMAIN/administrator	DOMAIN/domain users	client-pc (ipv4:192.0.2.1:57786)	SMB3_02		
-				AES-128-CMAC		

Service	pid	Machine	Connected at	Encryption	Signing:
---------	-----	---------	--------------	------------	----------

.....

example	969	192.0.2.1	Thu Nov 1 10:00:00 2018 CEST	-	AES-128-CMAC
---------	-----	-----------	------------------------------	---	--------------

Locked files:

Pid	Uid	DenyMode	Access	R/W	Oplock	SharePath	Name	Time
-----	-----	----------	--------	-----	--------	-----------	------	------

.....

969	10000	DENY_WRITE	0x120089	RDONLY	LEASE(RWH)	/srv/samba/example	file.txt	Thu Nov 1 10:00:00 2018
-----	-------	------------	----------	--------	------------	--------------------	----------	-------------------------

Additional resources

- **smbstatus(1)** man page

1.20.10. Using the smbtar utility

The **smbtar** utility backs up the content of an SMB share or a subdirectory of it and stores the content in a **tar** archive. Alternatively, you can write the content to a tape device.

Prerequisites

- The **samba-client** package is installed.

Procedure

- Use the following command to back up the content of the **demo** directory on the **//server/example/** share and store the content in the **/root/example.tar** archive:

```
# smbtar -s server -x example -u user_name -p password -t /root/example.tar
```

Additional resources

- **smbtar(1)** man page

1.20.11. Using the wbinfo utility

The **wbinfo** utility queries and returns information created and used by the **winbindd** service.

Prerequisites

- The **samba-winbind-clients** package is installed.

Procedure

You can use **wbinfo**, for example, to:

- List domain users:

```
# wbinfo -u
AD\administrator
AD\guest
...
```

- List domain groups:

```
# wbinfo -g
AD\domain computers
AD\domain admins
AD\domain users
...
```

- Display the SID of a user:

```
# wbinfo --name-to-sid="AD\administrator"
S-1-5-21-1762709870-351891212-3141221786-500 SID_USER (1)
```

- Display information about domains and trusts:

```
# wbinfo --trusted-domains --verbose
Domain Name  DNS Domain      Trust Type  Transitive  In  Out
BUILTIN      None            Yes         Yes  Yes
```

<i>server</i>		None	Yes	Yes	Yes		
<i>DOMAIN1</i>	<i>domain1.example.com</i>	None		Yes		Yes	Yes
<i>DOMAIN2</i>	<i>domain2.example.com</i>	External	No			Yes	Yes

Additional resources

- **wbinfo(1)** man page

1.21. ADDITIONAL RESOURCES

- **smb.conf(5)** man page
- **/usr/share/docs/samba-version/** directory contains general documentation, example scripts, and LDAP schema files, provided by the Samba project
- [Setting up Samba and the Clustered Trivial Database \(CTDB\) to share directories stored on an GlusterFS volume](#)
- [Mounting an SMB Share on Red Hat Enterprise Linux](#)

CHAPTER 2. EXPORTING NFS SHARES

As a system administrator, you can use the NFS server to share a directory on your system over network.

2.1. INTRODUCTION TO NFS

This section explains the basic concepts of the NFS service.

A Network File System (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables you to consolidate resources onto centralized servers on the network.

The NFS server refers to the **/etc/exports** configuration file to determine whether the client is allowed to access any exported file systems. Once verified, all file and directory operations are available to the user.

2.2. SUPPORTED NFS VERSIONS

This section lists versions of NFS supported in Red Hat Enterprise Linux and their features.

Currently, Red Hat Enterprise Linux 9 supports the following major versions of NFS:

- NFS version 3 (NFSv3) supports safe asynchronous writes and is more robust at error handling than the previous NFSv2; it also supports 64-bit file sizes and offsets, allowing clients to access more than 2 GB of file data.
- NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires an **rpcbind** service, supports Access Control Lists (ACLs), and utilizes stateful operations.

NFS version 2 (NFSv2) is no longer supported by Red Hat.

Default NFS version

The default NFS version in Red Hat Enterprise Linux 9 is 4.2. NFS clients attempt to mount using NFSv4.2 by default, and fall back to NFSv4.1 when the server does not support NFSv4.2. The mount later falls back to NFSv4.0 and then to NFSv3.

Features of minor NFS versions

Following are the features of NFSv4.2 in Red Hat Enterprise Linux 9:

Server-side copy

Enables the NFS client to efficiently copy data without wasting network resources using the **copy_file_range()** system call.

Sparse files

Enables files to have one or more *holes*, which are unallocated or uninitialized data blocks consisting only of zeroes. The **lseek()** operation in NFSv4.2 supports **seek_hole()** and **seek_data()**, which enables applications to map out the location of holes in the sparse file.

Space reservation

Permits storage servers to reserve free space, which prohibits servers to run out of space. NFSv4.2 supports the **allocate()** operation to reserve space, the **deallocate()** operation to unreserve space, and the **fallocate()** operation to preallocate or deallocate space in a file.

Labeled NFS

Enforces data access rights and enables SELinux labels between a client and a server for individual files on an NFS file system.

Layout enhancements

Provides the **layoutstats()** operation, which enables some Parallel NFS (pNFS) servers to collect better performance statistics.

Following are the features of NFSv4.1:

- Enhances performance and security of network, and also includes client-side support for pNFS.
- No longer requires a separate TCP connection for callbacks, which allows an NFS server to grant delegations even when it cannot contact the client: for example, when NAT or a firewall interferes.
- Provides exactly once semantics (except for reboot operations), preventing a previous issue whereby certain operations sometimes returned an inaccurate result if a reply was lost and the operation was sent twice.

2.3. THE TCP AND UDP PROTOCOLS IN NFSV3 AND NFSV4

NFSv4 requires the Transmission Control Protocol (TCP) running over an IP network.

NFSv3 could also use the User Datagram Protocol (UDP) in earlier Red Hat Enterprise Linux versions. In Red Hat Enterprise Linux 9, NFS over UDP is no longer supported. By default, UDP is disabled in the NFS server.

2.4. SERVICES REQUIRED BY NFS

This section lists system services that are required for running an NFS server or mounting NFS shares. Red Hat Enterprise Linux starts these services automatically.

Red Hat Enterprise Linux uses a combination of kernel-level support and service processes to provide NFS file sharing. All NFS versions rely on Remote Procedure Calls (RPC) between clients and servers. To share or mount NFS file systems, the following services work together depending on which version of NFS is implemented:

nfsd

The NFS server kernel module that services requests for shared NFS file systems.

rpcbind

Accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services can access them. The **rpcbind** service responds to requests for RPC services and sets up connections to the requested RPC service. This is not used with NFSv4.

rpc.mountd

This process is used by an NFS server to process **MOUNT** requests from NFSv3 clients. It checks that the requested NFS share is currently exported by the NFS server, and that the client is allowed to access it. If the mount request is allowed, the **nfs-mountd** service replies with a Success status and provides the File-Handle for this NFS share back to the NFS client.

rpc.nfsd

This process enables explicit NFS versions and protocols the server advertises to be defined. It works with the Linux kernel to meet the dynamic demands of NFS clients, such as providing server threads each time an NFS client connects. This process corresponds to the **nfs-server** service.

lockd

This is a kernel thread that runs on both clients and servers. It implements the Network Lock Manager (NLM) protocol, which enables NFSv3 clients to lock files on the server. It is started automatically whenever the NFS server is run and whenever an NFS file system is mounted.

rpc.statd

This process implements the Network Status Monitor (NSM) RPC protocol, which notifies NFS clients when an NFS server is restarted without being gracefully brought down. The **rpc-statd** service is started automatically by the **nfs-server** service, and does not require user configuration. This is not used with NFSv4.

rpc.rquotad

This process provides user quota information for remote users. The **rpc-rquotad** service, which is provided by the **quota-rpc** package, has to be started by user when the **nfs-server** is started.

rpc.idmapd

This process provides NFSv4 client and server upcalls, which map between on-the-wire NFSv4 names (strings in the form of **user@domain**) and local UIDs and GIDs. For **idmapd** to function with NFSv4, the **/etc/idmapd.conf** file must be configured. At a minimum, the **Domain** parameter should be specified, which defines the NFSv4 mapping domain. If the NFSv4 mapping domain is the same as the DNS domain name, this parameter can be skipped. The client and server must agree on the NFSv4 mapping domain for ID mapping to function properly.

Only the NFSv4 server uses **rpc.idmapd**, which is started by the **nfs-idmapd** service. The NFSv4 client uses the keyring-based **nfsidmap** utility, which is called by the kernel on-demand to perform ID mapping. If there is a problem with **nfsidmap**, the client falls back to using **rpc.idmapd**.

The RPC services with NFSv4

The mounting and locking protocols have been incorporated into the NFSv4 protocol. The server also listens on the well-known TCP port 2049. As such, NFSv4 does not need to interact with **rpcbind**, **lockd**, and **rpc-statd** services. The **nfs-mountd** service is still required on the NFS server to set up the exports, but is not involved in any over-the-wire operations.

Additional resources

- [Configuring an NFSv4 only server without **rpcbind**](#).

2.5. NFS HOST NAME FORMATS

This section describes different formats that you can use to specify a host when mounting or exporting an NFS share.

You can specify the host in the following formats:

Single machine

Either of the following:

- A fully-qualified domain name (that can be resolved by the server)
- Host name (that can be resolved by the server)
- An IP address.

IP networks

Either of the following formats is valid:

- **a.b.c.d/z**, where **a.b.c.d** is the network and **z** is the number of bits in the netmask; for

example **192.168.0.0/24**.

- **a.b.c.d/netmask**, where **a.b.c.d** is the network and **netmask** is the netmask; for example, **192.168.100.8/255.255.255.0**.

Netgroups

The **@group-name** format, where **group-name** is the NIS netgroup name.

2.6. NFS SERVER CONFIGURATION

This section describes the syntax and options of two ways to configure exports on an NFS server:

- Manually editing the **/etc/exports** configuration file
- Using the **exportfs** utility on the command line

2.6.1. The **/etc/exports** configuration file

The **/etc/exports** file controls which file systems are exported to remote hosts and specifies options. It follows the following syntax rules:

- Blank lines are ignored.
- To add a comment, start a line with the hash mark (**#**).
- You can wrap long lines with a backslash (****).
- Each exported file system should be on its own individual line.
- Any lists of authorized hosts placed after an exported file system must be separated by space characters.
- Options for each of the hosts must be placed in parentheses directly after the host identifier, without any spaces separating the host and the first parenthesis.

Export entry

Each entry for an exported file system has the following structure:

```
export host(options)
```

It is also possible to specify multiple hosts, along with specific options for each host. To do so, list them on the same line as a space-delimited list, with each host name followed by its respective options (in parentheses), as in:

```
export host1(options1) host2(options2) host3(options3)
```

In this structure:

export

The directory being exported

host

The host or network to which the export is being shared

options

The options to be used for host

Example 2.1. A simple `/etc/exports` file

In its simplest form, the `/etc/exports` file only specifies the exported directory and the hosts permitted to access it:

```
/exported/directory bob.example.com
```

Here, **bob.example.com** can mount `/exported/directory/` from the NFS server. Because no options are specified in this example, NFS uses default options.

IMPORTANT

The format of the `/etc/exports` file is very precise, particularly in regards to use of the space character. Remember to always separate exported file systems from hosts and hosts from one another with a space character. However, there should be no other space characters in the file except on comment lines.

For example, the following two lines do not mean the same thing:

```
/home bob.example.com(rw)
/home bob.example.com (rw)
```

The first line allows only users from **bob.example.com** read and write access to the `/home` directory. The second line allows users from **bob.example.com** to mount the directory as read-only (the default), while the rest of the world can mount it read/write.

Default options

The default options for an export entry are:

ro

The exported file system is read-only. Remote hosts cannot change the data shared on the file system. To allow hosts to make changes to the file system (that is, read and write), specify the `rw` option.

sync

The NFS server will not reply to requests before changes made by previous requests are written to disk. To enable asynchronous writes instead, specify the option **async**.

wdelay

The NFS server will delay writing to the disk if it suspects another write request is imminent. This can improve performance as it reduces the number of times the disk must be accessed by separate write commands, thereby reducing write overhead. To disable this, specify the **no_wdelay** option, which is available only if the default `sync` option is also specified.

root_squash

This prevents root users connected remotely (as opposed to locally) from having root privileges; instead, the NFS server assigns them the user ID **nobody**. This effectively "squashes" the power of the remote root user to the lowest local user, preventing possible unauthorized writes on the remote server. To disable root squashing, specify the **no_root_squash** option.

To squash every remote user (including root), use the **all_squash** option. To specify the user and group IDs that the NFS server should assign to remote users from a particular host, use the **anonuid** and **anongid** options, respectively, as in:

```
export host(anonuid=uid,anongid=gid)
```

Here, *uid* and *gid* are user ID number and group ID number, respectively. The **anonuid** and **anongid** options enable you to create a special user and group account for remote NFS users to share.

By default, access control lists (ACLs) are supported by NFS under Red Hat Enterprise Linux. To disable this feature, specify the **no_acl** option when exporting the file system.

Default and overridden options

Each default for every exported file system must be explicitly overridden. For example, if the **rw** option is not specified, then the exported file system is shared as read-only. The following is a sample line from **/etc/exports** which overrides two default options:

```
/another/exported/directory 192.168.0.3(rw,async)
```

In this example, **192.168.0.3** can mount **/another/exported/directory/** read and write, and all writes to disk are asynchronous.

2.6.2. The **exportfs** utility

The **exportfs** utility enables the root user to selectively export or unexport directories without restarting the NFS service. When given the proper options, the **exportfs** utility writes the exported file systems to **/var/lib/nfs/xtab**. Because the **nfs-mountd** service refers to the **xtab** file when deciding access privileges to a file system, changes to the list of exported file systems take effect immediately.

Common **exportfs** options

The following is a list of commonly-used options available for **exportfs**:

-r

Causes all directories listed in **/etc/exports** to be exported by constructing a new export list in **/var/lib/nfs/etab**. This option effectively refreshes the export list with any changes made to **/etc/exports**.

-a

Causes all directories to be exported or unexported, depending on what other options are passed to **exportfs**. If no other options are specified, **exportfs** exports all file systems specified in **/etc/exports**.

-o file-systems

Specifies directories to be exported that are not listed in **/etc/exports**. Replace *file-systems* with additional file systems to be exported. These file systems must be formatted in the same way they are specified in **/etc/exports**. This option is often used to test an exported file system before adding it permanently to the list of exported file systems.

-i

Ignores **/etc/exports**; only options given from the command line are used to define exported file systems.

-u

Unexports all shared directories. The command **exportfs -ua** suspends NFS file sharing while keeping all NFS services up. To re-enable NFS sharing, use **exportfs -r**.

-v

Verbose operation, where the file systems being exported or unexported are displayed in greater detail when the **exportfs** command is executed.

If no options are passed to the **exportfs** utility, it displays a list of currently exported file systems.

Additional resources

- [NFS host name formats](#) .

2.7. NFS AND RPCBIND

The **rpcbind** service maps Remote Procedure Call (RPC) services to the ports on which they listen. RPC processes notify **rpcbind** when they start, registering the ports they are listening on and the RPC program numbers they expect to serve. The client system then contacts **rpcbind** on the server with a particular RPC program number. The **rpcbind** service redirects the client to the proper port number so it can communicate with the requested service.

The Network File System Version 3 (NFSv3) requires the **rpcbind** service.

Because RPC-based services rely on **rpcbind** to make all connections with incoming client requests, **rpcbind** must be available before any of these services start.

Access control rules for **rpcbind** affect all RPC-based services. Alternatively, it is possible to specify access control rules for each of the NFS RPC daemons.

Additional resources

- **rpc.mountd(8)** man page
- **rpc.statd(8)** man page

2.8. INSTALLING NFS

This procedure installs all packages necessary to mount or export NFS shares.

Procedure

- Install the **nfs-utils** package:

```
# dnf install nfs-utils
```

2.9. STARTING THE NFS SERVER

This procedure describes how to start the NFS server, which is required to export NFS shares.

Prerequisites

- For servers that support NFSv3 connections, the **rpcbind** service must be running. To verify that **rpcbind** is active, use the following command:

```
$ systemctl status rpcbind
```

If the service is stopped, start and enable it:

```
$ systemctl enable --now rpcbind
```

Procedure

- To start the NFS server and enable it to start automatically at boot, use the following command:

```
# systemctl enable --now nfs-server
```

Additional resources

- [Configuring an NFSv4-only server.](#)

2.10. TROUBLESHOOTING NFS AND RPCBIND

Because the **rpcbind** service provides coordination between RPC services and the port numbers used to communicate with them, it is useful to view the status of current RPC services using **rpcbind** when troubleshooting. The **rpcinfo** utility shows each RPC-based service with port numbers, an RPC program number, a version number, and an IP protocol type (TCP or UDP).

Procedure

1. To make sure the proper NFS RPC-based services are enabled for **rpcbind**, use the following command:

```
# rpcinfo -p
```

Example 2.2. rpcinfo -p command output

The following is sample output from this command:

```
program vers proto  port  service
100000  4  tcp   111  portmapper
100000  3  tcp   111  portmapper
100000  2  tcp   111  portmapper
100000  4  udp   111  portmapper
100000  3  udp   111  portmapper
100000  2  udp   111  portmapper
100005  1  udp  20048 mountd
100005  1  tcp  20048 mountd
100005  2  udp  20048 mountd
100005  2  tcp  20048 mountd
100005  3  udp  20048 mountd
100005  3  tcp  20048 mountd
100024  1  udp  37769 status
100024  1  tcp  49349 status
100003  3  tcp   2049 nfs
100003  4  tcp   2049 nfs
100227  3  tcp   2049 nfs_acl
100021  1  udp  56691 nlockmgr
100021  3  udp  56691 nlockmgr
100021  4  udp  56691 nlockmgr
100021  1  tcp  46193 nlockmgr
100021  3  tcp  46193 nlockmgr
100021  4  tcp  46193 nlockmgr
```

If one of the NFS services does not start up correctly, **rpcbind** will be unable to map RPC requests from clients for that service to the correct port.

2. In many cases, if NFS is not present in **rpcinfo** output, restarting NFS causes the service to correctly register with **rpcbind** and begin working:

```
# systemctl restart nfs-server
```

Additional resources

- [Configuring an NFSv4-only server.](#)

2.11. CONFIGURING THE NFS SERVER TO RUN BEHIND A FIREWALL

NFS requires the **rpcbind** service, which dynamically assigns ports for RPC services and can cause issues for configuring firewall rules. The following sections describe how to configure NFS versions to work behind a firewall if you want to support:

- NFSv3
This includes any servers that support NFSv3:
 - NFSv3-only servers
 - Servers that support both NFSv3 and NFSv4
- NFSv4-only

2.11.1. Configuring the NFSv3-enabled server to run behind a firewall

The following procedure describes how to configure servers that support NFSv3 to run behind a firewall. This includes NFSv3-only servers and servers that support both NFSv3 and NFSv4.

Procedure

1. To allow clients to access NFS shares behind a firewall, configure the firewall by running the following commands on the NFS server:

```
firewall-cmd --permanent --add-service mountd
firewall-cmd --permanent --add-service rpc-bind
firewall-cmd --permanent --add-service nfs
```

2. Specify the ports to be used by the RPC service **nlockmgr** in the **/etc/nfs.conf** file as follows:

```
[lockd]

port=tcp-port-number
udp-port=udp-port-number
```

Alternatively, you can specify **nlm_tcpport** and **nlm_udpport** in the **/etc/modprobe.d/lockd.conf** file.

3. Open the specified ports in the firewall by running the following commands on the NFS server:

```

firewall-cmd --permanent --add-port=<lockd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<lockd-udp-port>/udp

```

4. Add static ports for **rpc.statd** by editing the **[statd]** section of the **/etc/nfs.conf** file as follows:

```

[statd]

port=port-number

```

5. Open the added ports in the firewall by running the following commands on the NFS server:

```

firewall-cmd --permanent --add-port=<statd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<statd-udp-port>/udp

```

6. Reload the firewall configuration:

```

firewall-cmd --reload

```

7. Restart the **rpc-statd** service first, and then restart the **nfs-server** service:

```

# systemctl restart rpc-statd.service
# systemctl restart nfs-server.service

```

Alternatively, if you specified the **lockd** ports in the **/etc/modprobe.d/lockd.conf** file:

- a. Update the current values of **/proc/sys/fs/nfs/nlm_tcpport** and **/proc/sys/fs/nfs/nlm_udpport**:

```

# sysctl -w fs.nfs.nlm_tcpport=<tcp-port>
# sysctl -w fs.nfs.nlm_udpport=<udp-port>

```

- b. Restart the **rpc-statd** and **nfs-server** services:

```

# systemctl restart rpc-statd.service
# systemctl restart nfs-server.service

```

2.11.2. Configuring the NFSv4-only server to run behind a firewall

The following procedure describes how to configure the NFSv4-only server to run behind a firewall.

Procedure

1. To allow clients to access NFS shares behind a firewall, configure the firewall by running the following command on the NFS server:

```

firewall-cmd --permanent --add-service nfs

```

2. Reload the firewall configuration:

```

firewall-cmd --reload

```

3. Restart the **nfs-server**:

```
# systemctl restart nfs-server
```

2.11.3. Configuring an NFSv3 client to run behind a firewall

The procedure to configure an NFSv3 client to run behind a firewall is similar to the procedure to configure an NFSv3 server to run behind a firewall.

If the machine you are configuring is both an NFS client and an NFS server, follow the procedure described in [Configuring the NFSv3-enabled server to run behind a firewall](#).

The following procedure describes how to configure a machine that is an NFS client only to run behind a firewall.

Procedure

1. To allow the NFS server to perform callbacks to the NFS client when the client is behind a firewall, add the **rpc-bind** service to the firewall by running the following command on the NFS client:

```
firewall-cmd --permanent --add-service rpc-bind
```

2. Specify the ports to be used by the RPC service **nlockmgr** in the **/etc/nfs.conf** file as follows:

```
[lockd]
port=port-number
udp-port=udp-port-number
```

Alternatively, you can specify **nlm_tcpport** and **nlm_udpport** in the **/etc/modprobe.d/lockd.conf** file.

3. Open the specified ports in the firewall by running the following commands on the NFS client:

```
firewall-cmd --permanent --add-port=<lockd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<lockd-udp-port>/udp
```

4. Add static ports for **rpc.statd** by editing the **[statd]** section of the **/etc/nfs.conf** file as follows:

```
[statd]
port=port-number
```

5. Open the added ports in the firewall by running the following commands on the NFS client:

```
firewall-cmd --permanent --add-port=<statd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<statd-udp-port>/udp
```

6. Reload the firewall configuration:

```
firewall-cmd --reload
```

7. Restart the **rpc-statd** service:


```
# systemctl restart rpc-statd.service
```

Alternatively, if you specified the **lockd** ports in the **/etc/modprobe.d/lockd.conf** file:

- a. Update the current values of **/proc/sys/fs/nfs/nlm_tcpport** and **/proc/sys/fs/nfs/nlm_udpport**:

```
# sysctl -w fs.nfs.nlm_tcpport=<tcp-port>
# sysctl -w fs.nfs.nlm_udpport=<udp-port>
```

- b. Restart the **rpc-statd** service:

```
# systemctl restart rpc-statd.service
```

2.11.4. Configuring an NFSv4 client to run behind a firewall

Perform this procedure only if the client is using NFSv4.0. In that case, it is necessary to open a port for NFSv4.0 callbacks.

This procedure is not needed for NFSv4.1 or higher because in the later protocol versions the server performs callbacks on the same connection that was initiated by the client.

Procedure

1. To allow NFSv4.0 callbacks to pass through firewalls, set **/proc/sys/fs/nfs/nfs_callback_tcpport** and allow the server to connect to that port on the client as follows:

```
# echo "fs.nfs.nfs_callback_tcpport = <callback-port>" >/etc/sysctl.d/90-nfs-callback-
port.conf
# sysctl -p /etc/sysctl.d/90-nfs-callback-port.conf
```

2. Open the specified port in the firewall by running the following command on the NFS client:

```
firewall-cmd --permanent --add-port=<callback-port>/tcp
```

3. Reload the firewall configuration:

```
firewall-cmd --reload
```

2.12. EXPORTING RPC QUOTA THROUGH A FIREWALL

If you export a file system that uses disk quotas, you can use the quota Remote Procedure Call (RPC) service to provide disk quota data to NFS clients.

Procedure

1. Enable and start the **rpc-rquotad** service:

```
# systemctl enable --now rpc-rquotad
```

**NOTE**

The **rpc-rquotad** service is, if enabled, started automatically after starting the **nfs-server** service.

2. To make the quota RPC service accessible behind a firewall, the TCP (or UDP, if UDP is enabled) port 875 need to be open. The default port number is defined in the **/etc/services** file. You can override the default port number by appending **-p port-number** to the **RPCRQUOTADOPTS** variable in the **/etc/sysconfig/rpc-rquotad** file.
3. By default, remote hosts can only read quotas. If you want to allow clients to set quotas, append the **-S** option to the **RPCRQUOTADOPTS** variable in the **/etc/sysconfig/rpc-rquotad** file.
4. Restart **rpc-rquotad** for the changes in the **/etc/sysconfig/rpc-rquotad** file to take effect:

```
# systemctl restart rpc-rquotad
```

2.13. ENABLING NFS OVER RDMA (NFSORDMA)

In Red Hat Enterprise Linux 9, Remote direct memory access (RDMA) service on RDMA-capable hardware provides Network File System (NFS) protocol support for high-speed file transfer over the network.

Procedure

1. Install the **rdma-core** package:

```
# dnf install rdma-core
```

2. Verify the lines with **xprtrdma** and **svcrdma** are not commented out in the **/etc/rdma/modules/rdma.conf** file:

```
# NFS over RDMA client support
xprtrdma
# NFS over RDMA server support
svcrdma
```

3. On the NFS server, create directory **/mnt/nfsordma** and export it to **/etc/exports**:

```
# mkdir /mnt/nfsordma
# echo "/mnt/nfsordma *(fsid=0,rw,async,insecure,no_root_squash)" >> /etc/exports
```

4. On the NFS client, mount the nfs-share with server IP address, for example, **172.31.0.186**:

```
# mount -o rdma,port=20049 172.31.0.186:/mnt/nfs-share /mnt/nfs
```

5. Restart the **nfs-server** service:

```
# systemctl restart nfs-server
```

Additional resources

- [The RFC 5667 standard](#)

2.14. ADDITIONAL RESOURCES

- [The Linux NFS wiki](#)

CHAPTER 3. SECURING NFS

To minimize NFS security risks and protect data on the server, consider the following sections when exporting NFS file systems on a server or mounting them on a client.

3.1. NFS SECURITY WITH AUTH_SYS AND EXPORT CONTROLS

NFS provides the following traditional options in order to control access to exported files:

- The server restricts which hosts are allowed to mount which file systems either by IP address or by host name.
- The server enforces file system permissions for users on NFS clients in the same way it does for local users. Traditionally, NFS does this using the **AUTH_SYS** call message (also called **AUTH_UNIX**), which relies on the client to state the UID and GIDs of the user. Be aware that this means that a malicious or misconfigured client might easily get this wrong and allow a user access to files that it should not.

To limit the potential risks, administrators often limits the access to read-only or squash user permissions to a common user and group ID. Unfortunately, these solutions prevent the NFS share from being used in the way it was originally intended.

Additionally, if an attacker gains control of the DNS server used by the system exporting the NFS file system, they can point the system associated with a particular hostname or fully qualified domain name to an unauthorized machine. At this point, the unauthorized machine *is* the system permitted to mount the NFS share, because no username or password information is exchanged to provide additional security for the NFS mount.

Wildcards should be used sparingly when exporting directories through NFS, as it is possible for the scope of the wildcard to encompass more systems than intended.

Additional resources

- To secure NFS and **rpcbind**, use, for example, **nftables** and **firewalld**.
- **nft(8)** man page
- **firewalld-cmd(1)** man page

3.2. NFS SECURITY WITH AUTH_GSS

All version of NFS support RPCSEC_GSS and the Kerberos mechanism.

Unlike AUTH_SYS, with the RPCSEC_GSS Kerberos mechanism, the server does not depend on the client to correctly represent which user is accessing the file. Instead, cryptography is used to authenticate users to the server, which prevents a malicious client from impersonating a user without having that user's Kerberos credentials. Using the RPCSEC_GSS Kerberos mechanism is the most straightforward way to secure mounts because after configuring Kerberos, no additional setup is needed.

3.3. CONFIGURING AN NFS SERVER AND CLIENT TO USE KERBEROS

Kerberos is a network authentication system that allows clients and servers to authenticate to each other by using symmetric encryption and a trusted third party, the KDC. Red Hat recommends using Identity Management (IdM) for setting up Kerberos.

Prerequisites

- The Kerberos Key Distribution Centre (**KDC**) is installed and configured.

Procedure

1. • Create the **nfs/hostname.domain@REALM** principal on the NFS server side.
 - Create the **host/hostname.domain@REALM** principal on both the server and the client side.
 - Add the corresponding keys to keytabs for the client and server.
2. On the server side, use the **sec=** option to enable the wanted security flavors. To enable all security flavors as well as non-cryptographic mounts:

```
/export *(sec=sys:krb5:krb5i:krb5p)
```

Valid security flavors to use with the **sec=** option are:

- **sys**: no cryptographic protection, the default
 - **krb5**: authentication only
 - **krb5i**: integrity protection
 - uses Kerberos V5 for user authentication and performs integrity checking of NFS operations using secure checksums to prevent data tampering.
 - **krb5p**: privacy protection
 - uses Kerberos V5 for user authentication, integrity checking, and encrypts NFS traffic to prevent traffic sniffing. This is the most secure setting, but it also involves the most performance overhead.
3. On the client side, add **sec=krb5** (or **sec=krb5i**, or **sec=krb5p**, depending on the setup) to the mount options:

```
# mount -o sec=krb5 server:/export /mnt
```

Additional resources

- [Creating files as root on krb5-secured NFS](#) . Not recommended.
- **exports(5)** man page
- **nfs(5)** man page

3.4. NFSV4 SECURITY OPTIONS

NFSv4 includes ACL support based on the Microsoft Windows NT model, not the POSIX model, because of the Microsoft Windows NT model's features and wide deployment.

Another important security feature of NFSv4 is the removal of the use of the **MOUNT** protocol for mounting file systems. The **MOUNT** protocol presented a security risk because of the way the protocol processed file handles.

3.5. FILE PERMISSIONS ON MOUNTED NFS EXPORTS

Once the NFS file system is mounted as either read or read and write by a remote host, the only protection each shared file has is its permissions. If two users that share the same user ID value mount the same NFS file system on different client systems, they can modify each others' files. Additionally, anyone logged in as root on the client system can use the **su -** command to access any files with the NFS share.

By default, access control lists (ACLs) are supported by NFS under Red Hat Enterprise Linux. Red Hat recommends to keep this feature enabled.

By default, NFS uses *root squashing* when exporting a file system. This sets the user ID of anyone accessing the NFS share as the root user on their local machine to **nobody**. Root squashing is controlled by the default option **root_squash**; for more information about this option, see [NFS server configuration](#).

When exporting an NFS share as read-only, consider using the **all_squash** option. This option makes every user accessing the exported file system take the user ID of the **nobody** user.

CHAPTER 4. ENABLING PNFS SCSI LAYOUTS IN NFS

You can configure the NFS server and client to use the pNFS SCSI layout for accessing data. pNFS SCSI is beneficial in use cases that involve longer-duration single-client access to a file.

Prerequisites

- Both the client and the server must be able to send SCSI commands to the same block device. That is, the block device must be on a shared SCSI bus.
- The block device must contain an XFS file system.
- The SCSI device must support SCSI Persistent Reservations as described in the SCSI-3 Primary Commands specification.

4.1. THE PNFS TECHNOLOGY

The pNFS architecture improves the scalability of NFS. When a server implements pNFS, the client is able to access data through multiple servers concurrently. This can lead to performance improvements.

pNFS supports the following storage protocols or layouts on RHEL:

- Files
- Flexfiles
- SCSI

4.2. PNFS SCSI LAYOUTS

The SCSI layout builds on the work of pNFS block layouts. The layout is defined across SCSI devices. It contains a sequential series of fixed-size blocks as logical units (LUs) that must be capable of supporting SCSI persistent reservations. The LU devices are identified by their SCSI device identification.

pNFS SCSI performs well in use cases that involve longer-duration single-client access to a file. An example might be a mail server or a virtual machine housing a cluster.

Operations between the client and the server

When an NFS client reads from a file or writes to it, the client performs a **LAYOUTGET** operation. The server responds with the location of the file on the SCSI device. The client might need to perform an additional operation of **GETDEVICEINFO** to determine which SCSI device to use. If these operations work correctly, the client can issue I/O requests directly to the SCSI device instead of sending **READ** and **WRITE** operations to the server.

Errors or contention between clients might cause the server to recall layouts or not issue them to the clients. In those cases, the clients fall back to issuing **READ** and **WRITE** operations to the server instead of sending I/O requests directly to the SCSI device.

To monitor the operations, see [Monitoring pNFS SCSI layouts functionality](#).

Device reservations

pNFS SCSI handles fencing through the assignment of reservations. Before the server issues layouts to clients, it reserves the SCSI device to ensure that only registered clients may access the device. If a client can issue commands to that SCSI device but is not registered with the device, many operations

from the client on that device fail. For example, the **blkid** command on the client fails to show the UUID of the XFS file system if the server has not given a layout for that device to the client.

The server does not remove its own persistent reservation. This protects the data within the file system on the device across restarts of clients and servers. In order to repurpose the SCSI device, you might need to manually remove the persistent reservation on the NFS server.

4.3. CHECKING FOR A SCSI DEVICE COMPATIBLE WITH PNFS

This procedure checks if a SCSI device supports the pNFS SCSI layout.

Prerequisites

- Install the **sg3_utils** package:

```
# dnf install sg3_utils
```

Procedure

- On both the server and client, check for the proper SCSI device support:

```
# sg_persist --in --report-capabilities --verbose path-to-scsi-device
```

Ensure that the *Persist Through Power Loss Active* (**PTPL_A**) bit is set.

Example 4.1. A SCSI device that supports pNFS SCSI

The following is an example of **sg_persist** output for a SCSI device that supports pNFS SCSI. The **PTPL_A** bit reports **1**.

```
inquiry cdb: 12 00 00 00 24 00
Persistent Reservation In cmd: 5e 02 00 00 00 00 00 20 00 00
LIO-ORG block11 4.0
Peripheral device type: disk
Report capabilities response:
Compatible Reservation Handling(CRH): 1
Specify Initiator Ports Capable(SIP_C): 1
All Target Ports Capable(ATP_C): 1
Persist Through Power Loss Capable(PTPL_C): 1
Type Mask Valid(TMV): 1
Allow Commands: 1
Persist Through Power Loss Active(PTPL_A): 1
Support indicated in Type mask:
Write Exclusive, all registrants: 1
Exclusive Access, registrants only: 1
Write Exclusive, registrants only: 1
Exclusive Access: 1
Write Exclusive: 1
Exclusive Access, all registrants: 1
```

Additional resources

- **sg_persist(8)** man page

4.4. SETTING UP PNFS SCSI ON THE SERVER

This procedure configures an NFS server to export a pNFS SCSI layout.

Procedure

1. On the server, mount the XFS file system created on the SCSI device.
2. Configure the NFS server to export NFS version 4.1 or higher. Set the following option in the **[nfsd]** section of the **/etc/nfs.conf** file:

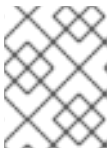
```
[nfsd]
vers4.1=y
```

3. Configure the NFS server to export the XFS file system over NFS with the **pnfs** option:

Example 4.2. An entry in **/etc/exports** to export pNFS SCSI

The following entry in the **/etc/exports** configuration file exports the file system mounted at **/exported/directory/** to the **allowed.example.com** client as a pNFS SCSI layout:

```
/exported/directory allowed.example.com(pnfs)
```



NOTE

The exported file system must be created on the whole block device, not only on a partition.

Additional resources

- [Exporting NFS shares](#).

4.5. SETTING UP PNFS SCSI ON THE CLIENT

This procedure configures an NFS client to mount a pNFS SCSI layout.

Prerequisites

- The NFS server is configured to export an XFS file system over pNFS SCSI. See [Setting up pNFS SCSI on the server](#).

Procedure

- On the client, mount the exported XFS file system using NFS version 4.1 or higher:

```
# mount -t nfs -o nfsvers=4.1 host:/remote/export /local/directory
```

Do not mount the XFS file system directly without NFS.

Additional resources

- [Mounting NFS shares.](#)

4.6. RELEASING THE PNFS SCSI RESERVATION ON THE SERVER

This procedure releases the persistent reservation that an NFS server holds on a SCSI device. This enables you to repurpose the SCSI device when you no longer need to export pNFS SCSI.

You must remove the reservation from the server. It cannot be removed from a different IT Nexus.

Prerequisites

- Install the **sg3_utils** package:

```
# dnf install sg3_utils
```

Procedure

1. Query an existing reservation on the server:

```
# sg_persist --read-reservation path-to-scsi-device
```

Example 4.3. Querying a reservation on /dev/sda

```
# sg_persist --read-reservation /dev/sda

LIO-ORG  block_1      4.0
Peripheral device type: disk
PR generation=0x8, Reservation follows:
Key=0x1000000000000000
scope: LU_SCOPE, type: Exclusive Access, registrants only
```

2. Remove the existing registration on the server:

```
# sg_persist --out \
    --release \
    --param-rk=reservation-key \
    --prout-type=6 \
    path-to-scsi-device
```

Example 4.4. Removing a reservation on /dev/sda

```
# sg_persist --out \
    --release \
    --param-rk=0x1000000000000000 \
    --prout-type=6 \
    /dev/sda

LIO-ORG  block_1      4.0
Peripheral device type: disk
```

■

Additional resources

- **sg_persist(8)** man page