pter 3. Server Types and Security Modes

Andrew Samba Team Tridgell

Samba Team

<tridge@samba.org>

Jelmer R. The Samba Team Vernooij

The Samba Team

<jelmer@samba.org>

John H. Samba Team Terpstra

Samba Team

<jht@samba.org>

Table of Contents

Features and Benefits
Server Types
Samba Security Modes

- User Level Security
- Share-Level Security
- Domain Security Mode (User-Level Security)
- ADS Security Mode (User-Level Security)
- Server Security (User Level Security)

Password Checking
Common Errors

- What Makes Samba a Server?
- What Makes Samba a Domain Controller?
- What Makes Samba a Domain Member?
- Constantly Losing Connections to Password Server
- Stand-alone Server is converted to Domain Controller Now User accounts don't work

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

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

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

Features and Benefits

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

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

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

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

- Samba-3 can replace an MS Windows NT4 domain controller.
- Samba-3 offers excellent interoperability with MS Windows NT4-style domains as well as natively with Microsoft Active Directory domains.
- Samba-3 permits full NT4-style interdomain trusts.
- Samba has security modes that permit more flexible authentication than is possible with MS Windows NT4 domain controllers.
- Samba-3 permits use of multiple concurrent account database backends. (Encrypted passwords that are stored in the account database are in formats that are unique to Windows networking).
- The account database backends can be distributed and replicated using multiple methods. This gives Samba-3 greater flexibility than MS Windows NT4 and in many cases a significantly higher utility than Active Directory domains with MS Windows 200x.

Server Types

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

- Domain Controller
  - Primary Domain Controller (PDC)
  - Backup Domain Controller (BDC)
  - ADS Domain Controller
- Domain Member Server
  - Active Directory Domain Server
  - NT4 Style Domain Domain Server
- Standalone Server

The chapters covering domain control (Domain Control), backup domain control (Backup Domain Control), and domain membership (Domain Membership) provide pertinent information regarding Samba configuration for each of these server roles. You are strongly encouraged to become intimately familiar with these chapters because they lay the foundation for deployment of Samba domain security.

A Standalone server is autonomous in respect of the source of its account backend. Refer to Standalone Servers to gain a wider appreciation of what is meant by a server being configured as a *standalone* server.

Samba Security Modes

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

Microsoft Windows networking uses a protocol that was originally called the Server Message Block (SMB) protocol. Since some time around 1996 the protocol has been better known as the Common Internet Filesystem (CIFS) protocol.

In the SMB/CIFS networking world, there are only two types of security:*user-level* and *share level*. We refer to these collectively as *security levels*. In implementing these two security levels, Samba provides flexibilities that are not available with MS Windows NT4/200x servers. In fact, Samba implements *share-level* security only one way, but has four ways of implementing *user-level* security. Collectively, we call the Samba implementations of the security levels *security modes*. They are known as*share*, *user*, *domain*, *ADS*, and *server* modes. They are documented in this chapter.

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

The term `client` refers to all agents whether it is a Windows workstation, a Windows server, another Samba server, or any vanilla SMB or CIFS client application (e.g., `smbclient`) that make use of services provided by an SMB/CIFS server.

User Level Security

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

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

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

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

Windows networking user account names are case-insensitive, meaning that upper-case and lower-case characters in the account name are considered equivalent. They are said to be case-preserving, but not case significant. Windows and LanManager systems previous to Windows NT version 3.10 have case-insensitive passwords that were not necessarily case-preserving. All Windows NT family systems treat passwords as case-preserving and case-sensitive.

Example Configuration

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

```
security = user
```

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

Share-Level Security

In share-level security, the client authenticates itself separately for each share. It sends a password along with each tree connection request (share mount), but it does not explicitly send a username with this operation. The client expects a password to be associated with each share, independent of the user. This means that Samba has to work out what username the client probably wants to use, because the username is not explicitly sent to the SMB server. Some commercial SMB servers such as NT actually associate passwords directly with shares in share-level security, but Samba always uses the UNIX authentication scheme where it is a username/password pair that is authenticated, not a share/password pair.

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

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

Where the list of possible user names is not provided, Samba makes a UNIX system call to find the user account that has a password that matches the one provided from the standard account database. On a system that has no name service switch (NSS) facility, such lookups will be from the `/etc/passwd` database. On NSS enabled systems, the lookup will go to the libraries that have been specified in the `nsswitch.conf` file. The entries in that file in which the libraries are specified are:

```
passwd: files nis ldap
shadow: files nis ldap
group: files nis ldap
```

In the example shown here (not likely to be used in practice) the lookup will check `/etc/passwd` and `/etc/group`, if not found it will check NIS, then LDAP.

Example Configuration

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

```
security = share
```

Domain Security Mode (User-Level Security)

Domain security provides a mechanism for storing all user and group accounts in a central, shared, account repository. The centralized account repository is shared between domain (security) controllers. Servers that act as domain controllers provide authentication and validation services to all machines that participate in the security context for the domain. A primary domain controller (PDC) is a server that is responsible for maintaining the integrity of the security account database. Backup domain controllers (BDCs) provide only domain logon and authentication services. Usually, BDCs will answer network logon requests more responsively than will a PDC.

When Samba is operating in security = domain mode, the Samba server has a domain security trust account (a machine account) and causes all authentication requests to be passed through to the domain controllers. In other words, this configuration makes the Samba server a domain member server, even when it is in fact acting as a domain controller. All machines that participate in domain security must have a machine account in the security database.

Within the domain security environment, the underlying security architecture uses user-level security. Even machines that are domain members must authenticate on startup. The machine account consists of an account entry in the accounts database, the name of which is the NetBIOS name of the machine and of which the password is randomly generated and known to both the domain controllers and the member machine. If the machine account cannot be validated during startup, users will not be able to log on to the domain using this machine because it cannot be trusted. The machine account is referred to as a machine trust account.

There are three possible domain member configurations:

1. Primary domain controller (PDC) - of which there is one per domain.
2. Backup domain controller (BDC) - of which there can be any number per domain.
3. Domain member server (DMS) - of which there can be any number per domain.

We will discuss each of these in separate chapters. For now, we are most interested in basic DMS configuration.

Example Configuration

Samba as a Domain Member Server

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

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

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

1. On the MS Windows NT domain controller, using the Server Manager, add a machine account for the Samba server.

2. On the UNIX/Linux system execute:

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

> **Note**
>
> Samba-2.2.4 and later Samba 2.2.x series releases can autojoin a Windows NT4-style domain just by executing:
>
> ```
> root# smbpasswd -j DOMAIN_NAME -r PDC_NAME \
>         -U Administrator%password
> ```
>
> Samba-3 can do the same by executing:
>
> ```
> root# net rpc join -U Administrator%password
> ```
>
> It is not necessary with Samba-3 to specify the *DOMAIN_NAME* or the *PDC_NAME*, as it figures this out from the `smb.conf` file settings.

Use of this mode of authentication requires there to be a standard UNIX account for each user in order to assign a UID once the account has been authenticated by the Windows domain controller. This account can be blocked to prevent logons by clients other than MS Windows through means such as setting an invalid shell in the `/etc/passwd` entry. The best way to allocate an invalid shell to a user account is to set the shell to the file`/bin/false`.

Domain controllers can be located anywhere that is convenient. The best advice is to have a BDC on every physical network segment, and if the PDC is on a remote network segment the use of WINS (see Network Browsing for more information) is almost essential.

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

For more information regarding domain membership, Domain Membership.

ADS Security Mode (User-Level Security)

Both Samba-2.2, and Samba-3 can join an Active Directory domain using NT4 style RPC based security. This is possible if the domain is run in native mode. Active Directory in native mode perfectly allows NT4-style domain members. This is contrary to popular belief.

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

Sites that use Microsoft Windows active directory services (ADS) should be aware of the significance of the terms: `native mode` and `mixed mode` ADS operation. The term `realm` is used to describe a Kerberos-based security architecture (such as is used by Microsoft ADS).

Example Configuration

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

The following parameter may be required:

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

Please refer to Domain Membership, and Samba ADS Domain Membershipfor more information regarding this configuration option.

Server Security (User Level Security)

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

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

In server security mode the Samba server reports to the client that it is in user-level security. The client then does a session setup as described earlier. The Samba server takes the username/password that the client sends and attempts to log into the password server by sending exactly the same username/password that it got from the client. If that server is in user-level security and accepts the password, then Samba accepts the client's connection. This parameter allows the Samba server to use another SMB server as the password server.

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

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

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

Example Configuration

Using MS Windows NT as an Authentication Server

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

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

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

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

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

Password Checking

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

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

- An MD4 hash of the unicode of the password string. This is known as the NT hash.
- The password is converted to uppercase, and then padded or truncated to 14 bytes. This string is then appended with 5 bytes of NULL characters and split to form two 56-bit DES keys to encrypt a "magic" 8-byte value. The resulting 16 bytes form the LanMan hash.

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

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

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

The following parameters can be used to work around the issue of Windows 9x/Me clients

uppercasing usernames and passwords before transmitting them to the SMB server when using clear-text authentication:

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

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

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

Common Errors

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

Here we look at common mistakes and misapprehensions that have been the subject of discussions on the Samba mailing lists. Many of these are avoidable by doing your homework before attempting a Samba implementation. Some are the result of a misunderstanding of the English language, which has many phrases that are potentially vague and may be highly confusing to those for whom English is not their native tongue.

What Makes Samba a Server?

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

Samba is a server regardless of which security mode is chosen. When Samba is used outside of a domain security context, it is best to leave the security mode at the default setting. By default Samba-3 uses user-mode security.

What Makes Samba a Domain Controller?

The `smb.conf` parameter security = domain does not really make Samba behave as a domain controller. This setting means we want Samba to be a domain member. See Samba as a PDC for more information.

What Makes Samba a Domain Member?

Guess! So many others do. But whatever you do, do not think that security = user makes Samba act as a domain member. Read the manufacturer's manual before the warranty expires.

See Domain Membership, for more information.

Constantly Losing Connections to Password Server

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

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

Stand-alone Server is converted to Domain Controller Now User accounts don't work

“ When I try to log in to the DOMAIN, the eventlog shows *tried credentials DOMAIN/username; effective credentials SERVER/username* ”

Usually this is due to a user or machine account being created before the Samba server is configured to be a domain controller. Accounts created before the server becomes a domain controller will be *local* accounts and authenticated as what looks like a member in the SERVER domain, much like local user accounts in Windows 2000 and later. Accounts created after the Samba server becomes a domain controller will be *domain* accounts and will be authenticated as a member of the DOMAIN domain.

This can be verified by issuing the command `pdbedit -L -v username`. If this reports DOMAIN then the account is a domain account, if it reports SERVER then the account is a local account.

The easiest way to resolve this is to remove and recreate the account; however this may cause problems with established user profiles. You can also use `pdbedit -u username -I DOMAIN`. You may also need to change the User SID and Primary Group SID to match the domain.