Chapter 3

SERVER TYPES AND SECURITY MODES

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

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

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

3.1 Features and Benefits

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

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

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

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

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

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

3.2 Server Types

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

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

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

3.3 Samba Security Modes

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

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

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

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

3.3.1 User Level Security

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

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

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

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

3.3.1.1 Example Configuration

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

```
security = user
```

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

3.3.2 Share Level Security

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

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

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

3.3.2.1 Example Configuration

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

```
security = share
```

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

3.3.3 Domain Security Mode (User Level Security)

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

3.3.3.1 Example Configuration

Samba as a Domain Member Server

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

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

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

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

2. On the UNIX/Linux system execute:

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

NOTE

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

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

Samba-3 can do the same by executing:

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

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

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

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

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

3.3.4 ADS Security Mode (User Level Security)

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

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

3.3.4.1 Example Configuration

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

The following parameter may be required:

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

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

3.3.5 Server Security (User Level Security)

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

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

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

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

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

NOTE

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

3.3.5.1 Example Configuration

Using MS Windows NT as an Authentication Server

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

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

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

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

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

3.4 Password Checking

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

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

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

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

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

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

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

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

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

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

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

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

3.5 Common Errors

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

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

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

3.5.1 What Makes Samba a Server?

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

3.5.2 What Makes Samba a Domain Controller?

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

3.5.3 What Makes Samba a Domain Member?

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

3.5.4 Constantly Losing Connections to Password Server

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

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