NTP Administration Guide

Novell®
NetWare®

6.5 SP8

November 9, 2009

Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.

Contents

About this Guide

This guide describes the time synchronization service based on RFC 1305. Time synchronization uses the Network Time Protocol version 3 (NTPv3). This guide is intended to help network administrators configure and use NTPv3, and it contains the following sections:

Audience

This Guide is intended for NetWare administrators who use NTPv3 for Time synchronization services on NetWare 6.5.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to Novell documentation feedback (http://www.novell.com/documentation/feedback.html) and enter your comments there.

Documentation Updates

For the most recent version of this *Network Time Protocol Administration Guide*, see the Novell Documentation Web site (http://www.novell.com/documentation/lg/nw65).

Additional Documentation

For documentation on Time synchronization on NetWare®, see .

Documentation Conventions

All occurrences of NTP in this documentation refer to NTP version 3 (NTPv3).

In this documentation, a greater-than symbol (>) is used to separate actions within a step and items in a cross-reference path.

Also, a trademark symbol ($^®$, ™, etc.) denotes a Novell$^®$ trademark. An asterisk (*) denotes a third-party trademark.

When a single pathname can be written with a backslash for some platforms or a forward slash for other platforms, the pathname is presented with a backslash. Users of platforms that require a forward slash, such as UNIX*, should use forward slashes as required by your software.

Network Time Protocol

<div align="right">1</div>

The Network Time Protocol (NTP) is used to synchronize servers that have NTPv3 compliant operating systems like NetWare®, Linux*, and Solaris.*

This section explains the following:

For more information on NTP, refer to the following Web sites:

- NTP Home (http://www.ntp.org)
- Prof.Mills Home page (http://www.eecis.udel.edu/~mills)

1.1 NTP Terminology

This documentation uses the following terms:

- **NTP time provider:** A server that uses the Network Time Protocol (NTP) protocol and provides NTP time to other servers or to workstations on the network.

  The time provider gives time to operating systems that are NTP and NCP☐ compliant, such as the following:

  - NetWare 4.2, 5.0, 5.1, 6.0, and 6.5
  - All flavors of UNIX
  - All versions of Windows* that have NTP compliance

  The time provider can broadcast or multicast its services on the network. For more information, see Section 3.3, "Broadcast and Multicast Mode," on page 16.

- **NTP time consumer:** A server that understands the NTP protocol and seeks NTP time from an NTP time provider to synchronize its time.

  The time consumer can accept a time provider in the client-server, peer-peer, and multicast/ broadcast modes. For more information, see Section 3.1, "Client-Server Mode," on page 15, Section 3.2, "Peer-to-Peer Mode," on page 16, and Section 3.3, "Broadcast and Multicast Mode," on page 16.

  The time consumer can work with operating systems that are NTP and NCPcompliant, such as the following:

  - NetWare 5.1, 6.0, and 6.5
  - All flavors of UNIX
  - All versions of Windows that have NTP compliance

- **Time provider group:** A set of servers that are configured to ensure fault tolerance and optimal network usage.

The time provider group can be configured to keep the network traffic at a minimum.

 ◆ **Dispersion:** A measure (in seconds) of how scattered the time offsets are from a given time server.

 ◆ **Drift:** A measure (in hertz per second) of how quickly the skew of a clock changes.

   See also "Slew:" on page 10.

 ◆ **Jitter:** Small rapid variations in a waveform because of fluctuations in the voltage supply, mechanical vibrations, or other sources.

 ◆ **Minpoll:** Specifies the minimum polling interval (in seconds to the power of 2) for NTP messages. If you set minpoll to 4, the minimum polling interval reduces. A value of 4 with minpoll helps the server to synchronize within a minute.

   The default minpoll value is 4 on NetWare and 6 on UNIX.

 ◆ **Root delay:** The total round trip delay (in seconds) to the primary reference source at the root of the synchronization subnet. This variable can take on both positive and negative values, depending on clock precision and skew.

 ◆ **Root dispersion:** The maximum error (in seconds) relative to the primary reference source at the root of the synchronization subnet. This variable can take only positive values greater than zero.

 ◆ **Skew:** A measure (in hertz) of the difference between the actual frequency of a clock and its frequency to keep perfect time.

   See also "Drift:" on page 10.

 ◆ **Slam:** To immediately correct or adjust the time of a clock. This might lead to sudden bursts in time.

 ◆ **Step:** To change the time of a clock to the correct time with no intermediate adjustments.

   See "Skew:" on page 10.

 ◆ **Slew:** To gradually adjust the time of a clock until it displays the correct time.

   See "Step:" on page 10.

 ◆ **Stratum:** Conventions established to indicate the accuracy of each time server, defined by a number called the stratum, with the topmost level (PRIMARY Servers) assigned as one and each level downwards (SECONDARY Servers) in the hierarchy assigned as one greater than the preceding level.

1.2 NTP Implementation on NetWare

Novell® eDirectory™ 8.7.3 and above, relies heavily on consistent and reliable time stamps for its objects. Time stamps are used in synchronization of directories.

By default, Timesync is loaded with NetWare.

NTP addresses fault tolerance by using a time provider group. In a time provider group, all the servers in one geographical location network obtain time from other servers in the same network. Only one server communicates with a server outside the network and obtains time from it. Therefore, the network traffic across the geographical locations is reduced minimizing traffic across routers and WANs.

Features of NTP on NetWare

◆ All features are as per RFC 1305 on NTPv3.

◆ Supports for browser-based configuration Novell Remote Manager. This helps in centralizing the support and configuration of the NTP time synchronization service on the network for the complete eDirectory tree.

◆ Supports backward compatibility for Timesync.

1.3 NTP Architecture

This section briefly outlines the NTP architecture.

Figure 1-1 *NTP Architecture*
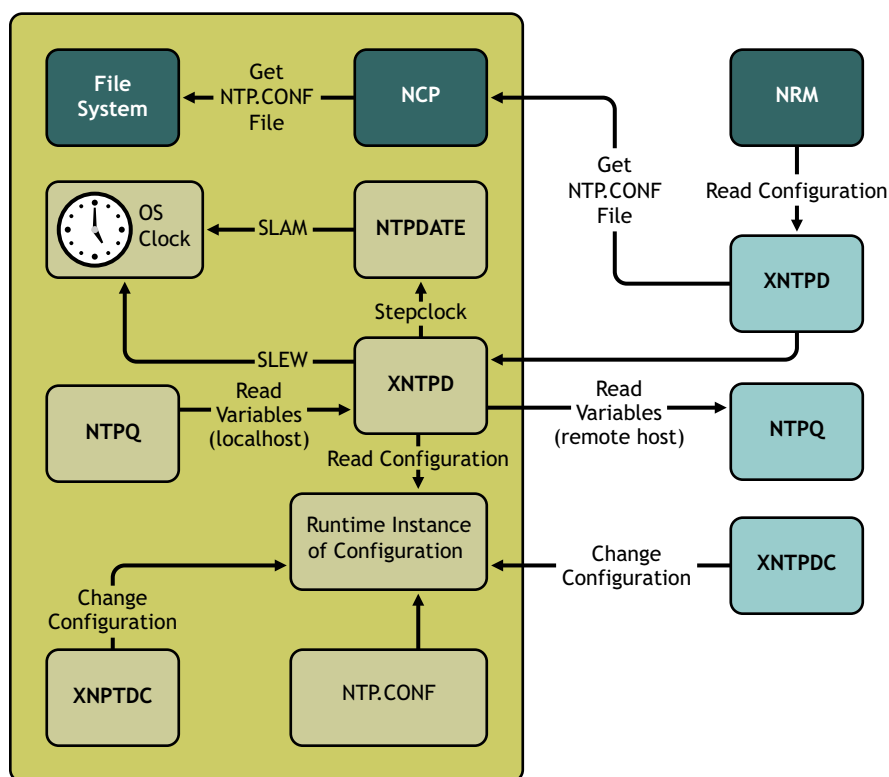
The above figure illustrates the following:

◆ **NTPDate:** Sets the local time and date.

◆ **NTPQ:** Queries the status or quality of time parameters.

◆ **NTPTrace:** Queries the time server and its servers until the master server is queried. NTPTrace determines where a given NTP server gets its time from, then follows the chain of NTP servers back to their master time source.

◆ **XNTPDC:** The remote configuration utility. It is used to query the XNTPD daemon about its current state and to request changes in that state.

◆ **XNTPD:** An operating system daemon that sets and maintains the system time of day in synchronization with Internet standard time servers.

NOTE: To make XNTPD to be loaded with NetWare by default, edit the `sys:\system\timeserv.ncf` file.

The components within the box relate to a particular server.

NTPDate and XNTPD communicate with the server's XNTPD, which is trying to synchronize the time.

NTPDate gets the time from another server and slams the time on the local clock. Slamming the time immediately overwrites the time on the local clock.

XNTPD gets the time from another server and slews the time on the local clock. Slewing the time gradually adjusts the local clock to the time of the other server. XNTPD sets and maintains the time on the local clock.

`ntp.conf` is the configuration file. XNTPD reads this file at startup in order to determine the synchronization sources and operating modes. The time configuration values are entered in the `ntp.conf` file.

1.4 Compatibility with Other Versions of NetWare

NTPv3 components cannot be loaded on a server where `timesync.nlm` is running.

All versions of NetWare 4 can use only NCP. NetWare 5.*x* and 6.*x* can use NCP and NTP.

The NTPv3 component, XNTPD, replies to NCP time requests to support backward compatibility for Timesync.

Table 1-1 *Compatibility Between NTP And Timesync*

Time Provider	Time Consumer	Is Configuration Allowed?
Timesync SINGLE time server	NTP client time consumer	Allowed - NTP
Timesync PRIMARY time server	NTP client time consumer	Allowed - NTP
Timesync SECONDARY time server	NTP client time consumer	Allowed - NTP
NTP server time provider	SINGLE time server	Allowed - NCP/NTP
	REFERENCE time server	Allowed - NCP/NTP
	PRIMARY time server	Allowed - NCP/NTP
	SECONDARY time server	Allowed - NCP/NTP
	NTP client Time Consumer	NTP

Running NTPv3 in a Virtualized Environment

2

Novell NTPv3 runs in a Xen virtualized environment just as it does on a physical NetWare server, or on a physical server running OES 2 Linux, and requires no special configuration or other changes.

To get started with virtualization, see "Introduction to Xen Virtualization" (http://www.novell.com/documentation/sles10/book_virtualization_xen/data/sec_xen_basics.html) in the *Virtualization with Xen* (http://www.novell.com/documentation/sles10/book_virtualization_xen/data/book_virtualization_xen.html) guide.

For information on setting up virtualized NetWare, see "Installing and Managing NetWare on a Xen-based VM" in the *OES 2 SP2: Installation Guide*.

For information on setting up virtualized OES 2 Linux, see "Installing, Upgrading, or Updating OES on a Xen-based VM" in the *OES 2 SP2: Installation Guide*.

2.1 What's Next

Once you are done with these preliminaries, you can start migrating to NetWare for NTPv3. To get started, continue with

Modes of Time Synchronization

<div style="text-align: right; font-size: xx-large;">3</div>

You can synchronize time using the following three methods:

3.1 Client-Server Mode

In this mode, the time consumer requests the time provider for the time and the time provider resplies back with the time, taking into account the time delays and other contingencies.

Figure 3-1   *Client-Server Mode*
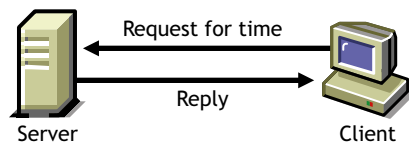
The time provider might also be a time consumer to another time provider. This scenario is displayed in the next figure. In this case, the time provider (Server1) requests the time from its time provider (Server2) and, upon getting a reply, responds to the time consumer (Client).

Figure 3-2   *Time Provider As a Time Consumer*
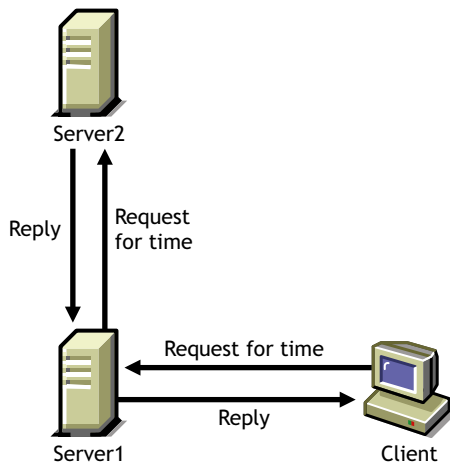
You must synchronize the server prior to synchronizing the client. Therefore, it is important to configure the server in either of the following ways:

- **Self-synchronized:** Here, the local clock is used as the time source and the server synchronizes with it. Add the following lines to the `ntp.conf` file (located in `sys:\etc`)

```
server 127.127.1.0
```

```
fudge server 127.127.1.0 stratum 3
```

- **As a client to another server:** Here, the server is configured as a time consumer to another time provider. Add the following line in the `ntp.conf` file:

  ```
  server IP_address_of_time_provider
  ```

  After you configure the server, you can configure the clients to use this server as a time provider. To do this, add the following lines to the `ntp.conf` file:

  ```
  server IP_address_of_server
  ```

See Section 4.1, "Manual Configuration," on page 19 for more information.

3.2 Peer-to-Peer Mode

In this mode, there are two time consumers or time providers. Both of them can request the time from each other and respond to each other. They are at the same level and therefore, known as peers.

Figure 3-3 *Peer-to-Peer Mode*
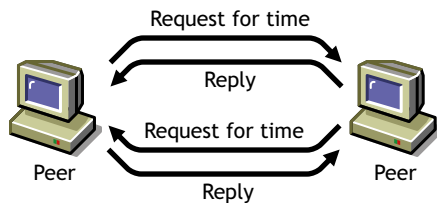
If you want to use the servers in this mode, add the following line to the `ntp.conf` file of each server:

```
peer IP_address_of_the_other_server
```

See Section 4.3, "Wide Area Configuration," on page 24 for more information.

3.3 Broadcast and Multicast Mode

In the broadcast or multicast mode, the time provider broadcasts (or multicasts) its service within the subnet. The time consumer listens to the broadcast (or multicast) and registers it as its time provider. The time consumer then requests the time from the time provider.

This mode helps to avoid reconfiguring the entire network if the time provider of the network changes.

To change the time provider, you must remove the time provider from the network and replace it with another to broadcast (or multicast) its service.

Figure 3-4 *Broadcast/Multicast Mode*
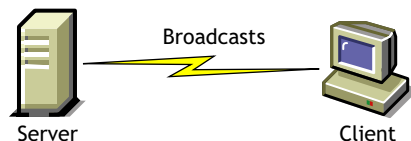
To configure servers in the broadcast or multicast mode, you must have a time provider advertising its service. The time consumers that need to obtain time from this time provider listen to the advertisement, register for the service, and use it.

3.3.1 Broadcast

To make the time provider broadcast its service, add the following line to its `ntp.conf` file:

```
broadcast subnet_broadcast_address key key_ID
```

To make the time consumer listen to the services that are broadcast, add the following line to its `ntp.conf` file:

```
broadcastclient subnet_broadcast_address
```

NOTE: The *subnet broadcast address* variable is optional.

3.3.2 Multicast

To make the time provider multicast its service, add the following line to its `ntp.conf` file:

```
broadcast 224.0.1.1 key key_ID
```

To make the time consumer listen to the services that are multicast, add the following line to its `ntp.conf` file:

```
multicastclient 224.0.1.1
```

See Section 4.2, "Auto Configuration," on page 22 for more details.

NTP Configuration

4

You can configure NTPv3 as a replacement for `timesync.nlm`.

Prior to configuring NTP, it is important to understand the different types of time synchronization methods. The NTP configuration is based on the synchronization type you choose to use.

There are three types of configuration methods:

It is important to understand when to use a particular type of configuration. The following table explains the type of configuration you can select for each mode of synchronization:

Table 4-1   *Configuration Type and Synchronization Mode*

Configuration Type	Synchronization Mode
Manual	◆ Client-server ◆ Self-synchronized
Auto	Broadcast/Multicast
Wide Area	◆ Peer-to-Peer ◆ A combination of modes

4.1 Manual Configuration

Manual configuration is easy to plan, configure, and debug. This type of configuration is best suited in a setup where there are fewer than 15 servers and the servers are in the same geographical location and do not span across a large WAN.

You can manually configure a time synchronized setup by completing the following tasks:

4.1.1 Planning the Setup

You should plan the setup before configuring the time provider and time consumers. The setup consists of a time group, which is a set of servers synchronized for time.

The plan should include the following:

- Identify the most reliable server in the subnet and make it the time provider.
- Identify the other servers in the subnet be the time consumers.

4.1.2 Configuring the Time Providers

In manual configuration, a time provider can get time from the following:

- From another time provider "Client-Server Mode" on page 20
- From its local clock "Self-Synchronized Mode" on page 20

Client-Server Mode

In this mode, a time consumer takes time from a time provider. The time provider can be a time consumer in another setup or can take time from an external time provider as shown in the figure below.

Figure 4-1   *Time Group Taking Time from a Time Provider*
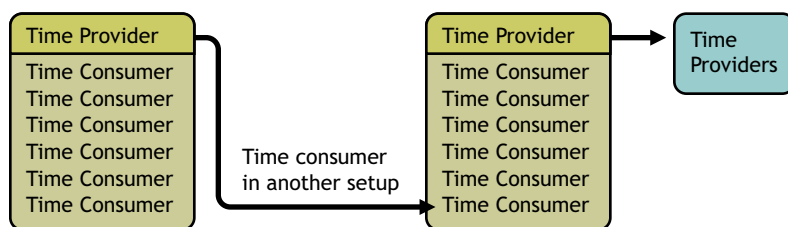
To configure the time provider:

1 Add a line similar to the following to the time provider's `ntp.conf` file:

`server IP_address_of_time_provider prefer`

The prefer parameter marks the server as preferred. All other things being equal, this time provider is chosen for synchronization among a set of correctly operating providers.

`ntp.conf` file is the configuration file for NTP. This file is located in `sys:\etc`. For more information, see "The Configuration File" on page 46.

2 Load XNTPD for the changes to take effect.

To do this, enter the following at the command prompt:

`Load XNTPD`

For more information, see Section 3.1, "Client-Server Mode," on page 15.

Self-Synchronized Mode

In this mode, the server takes time from its own local clock as shown in this figure.

Figure 4-2 *Time Group with Self-Synchronization*
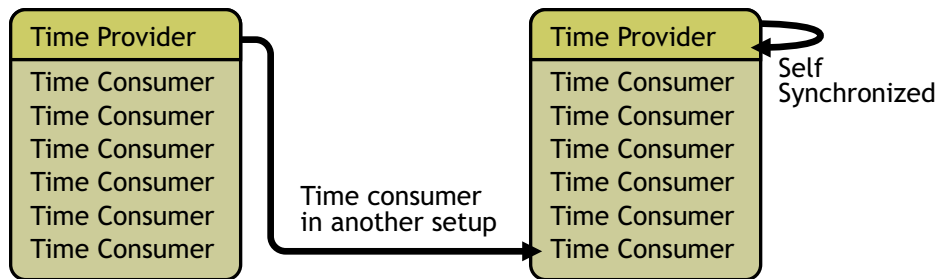
To configure the time provider:

1 Add lines similar to the following to the time provider's `ntp.conf` file:

```
server 127.127.1.0 prefer
fudge 127.127.1.0 stratum 3
```

2 Load XNTPD for the changes to take effect.

To do this, enter the following at the command prompt:

```
Load XNTPD
```

4.1.3 Configuring the Time Consumers

1 Add a line similar to the following to the time consumer's `ntp.conf` file:

```
server IP_address_of_time_provider prefer
```

2 Load XNTPD for the changes to take effect.

To do this, enter the following at command prompt:

```
Load XNTPD
```

4.1.4 Sample Scenario

This sample scenario in Figure 4-3 on page 22 demonstrates how to configure a time-synchronized setup in the manual mode.

Figure 4-3 *Sample Scenario for Manual Configuration*
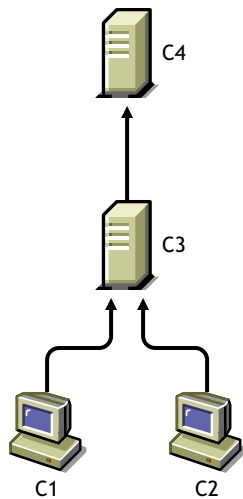
In this scenario:

- C1 and C2 are time consumers that obtain time from the time provider C3.
- C3 is also a time consumer that obtains time from the time provider C4.
- C4 is self-synchronized.

To use manual configuration to configure the setup explained in the scenario:

1 In the `ntp.conf` files of C1 and C2, add a line similar to the following:

   `server IP_address_of_C3 prefer`

2 In the `ntp.conf` file of C3, add a line similar to the following:

   `server IP_address_of_C4 prefer`

3 In the `ntp.conf` file of C4, add a line similar to the following:

   `server 127.127.1.0 prefer`

   `fudge 127.127.1.0 stratum 3`

4 Load XNTPD for the changes to take effect.

   To do this, enter the following at the command prompt:

   `Load XNTPD`

4.2 Auto Configuration

Auto configuration is best suited for setups where the time provider is expected to change frequently because, it does not require the time consumer to be reconfigured when the time provider changes.

Thus, the setup can be reconfigured in a single step without modifying the configuration of the time consumers in the network.

You can configure a time-synchronized setup using auto configuration by completing the following tasks:

- Section 4.2.1, "Planning the Setup," on page 23

4.2.1 Planning the Setup

You must plan the setup to identify a time provider. Identify the most reliable server in the subnet and make it the broadcast or multicast server.

4.2.2 Configuring the Time Provider

Broadcast

Add the following line in the time provider's `ntp.conf` file (located in `sys:\etc` to broadcast the time synchronization service on the network:

```
broadcast subnet_broadcast_address key key_ID
```

Multicast

Add the following line in the time provider's `ntp.conf` file (located in `sys:\etc` to multicast the time synchronization service on the network:

```
broadcast 224.0.1.1 key key_ID
```

4.2.3 Configuring the Time Consumers

Broadcast

To make the time consumer listen to the broadcast of the time provider's time synchronization service, add the following line to its `ntp.conf` file.

```
broadcastclient subnet_broadcast_address
```

NOTE: The *subnet broadcast address* variable is optional.

Multicast

To make the time consumer listen to the multicast of the time provider's time synchronization service, add the following line to its `ntp.conf` file.

```
multicastclient
```

4.2.4 Sample Scenario

This sample scenario in the following figure demonstrates how to configure a setup using auto configuration.

Figure 4-4 *Sample Scenario for Auto Configuration*
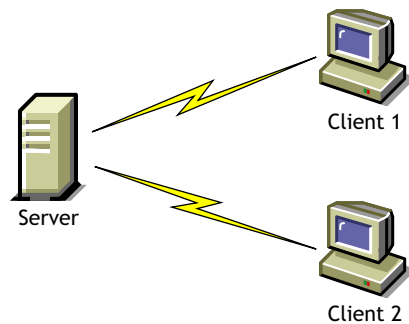
In this scenario:

- Server1 broadcasts its time synchronization service to all the time consumers in the subnet
- Client1 and Client2 are two time consumers in the subnet that listen to the broadcast

To use auto configuration to configure the setup in this scenario:

1 Add the following line to the `ntp.conf` file of Server1:

   `broadcast` *`subnet_broadcast_address`* `key` *`key_ID`*

2 Add the following line to the `ntp.conf` file of Client1 and Client2:

   `broadcastclient` *`subnet_broadcast_address`*

3 Load XNTPD for the changes to take effect.

   To do this, enter the following at the command prompt:

   `Load XNTPD`

4.3 Wide Area Configuration

Wide area configuration is best suited for setups where the servers are spread across geographical locations. It is also suitable for setups that need fault tolerance.

Wide area configuration can be achieved by completing the following tasks:

- Section 4.3.1, "Planning the Setup," on page 24
- Section 4.3.2, "Configuring the Time Provider Group," on page 25
- Section 4.3.3, "Configuring the Time Consumers," on page 25
- Section 4.3.4, "Sample Scenario," on page 26

4.3.1 Planning the Setup

Create a plan for configuring the time provider group, which is a set of servers configured to ensure fault tolerance and optimal network usage.

4.3.2 Configuring the Time Provider Group

A time provider group consists of a set of servers that synchronize time in a fault tolerance setup and minimize network traffic.

- "Setting Fault Tolerance" on page 25
- "Minimizing Network Traffic" on page 25

Setting Fault Tolerance

1 Configure at least two servers to communicate with each other in the peer-to-peer mode by adding a line similar to the following to each server's `ntp.conf` file:

```
peer IP_address_of_peer
```

2 Configure the servers to contact their own unique external time sources in the client-server mode by adding a line similar to the following to each server's `ntp.conf` file:

```
server IP_address_of_own_external_time_source
```

If one external time source link goes down, both time providers do not lose time synchronization.

3 Configure the servers to fall back to their local clocks (self-synchronize) and ensure that the external time source gets preference over the local clock. To achieve this, use a lower preference for the local clock (stratum value of 3). Add lines similar to the following to each server's `ntp.conf` file:

```
server 127.127.1.0
fudge 127.127.1.0 stratum 3
```

NOTE: It is recommended to use a higher value (maximum 16) for local clocks.

Minimizing Network Traffic

To minimize network traffic, configure two servers, on either side of the network, in either the peer-to-peer mode or the client-server mode. These two servers can either have their own external sources or be self-synchronized.

For more information, see Section 3.2, "Peer-to-Peer Mode," on page 16 and Section 3.1, "Client-Server Mode," on page 15.

4.3.3 Configuring the Time Consumers

- "Setting Fault Tolerance" on page 25
- "Minimizing Network Traffic" on page 26

Setting Fault Tolerance

To set fault tolerance, configure all the time consumers to have at least two time providers either in the client-server mode or the broadcast/multicast mode.

For more information, see Section 3.1, "Client-Server Mode," on page 15 and Section 3.3, "Broadcast and Multicast Mode," on page 16.

Minimizing Network Traffic

To minimize network traffic, time consumers should not contact the time providers that are across costly WANs. Preferably, a time consumer should contact a time provider within its own local network.

You can use either manual configuration or auto configuration to configure a time consumer.

To use manual configuration, add lines similar to the following to each time consumer's `ntp.conf` file:

```
server IP_address_of_time_provider1_within_same_network
```

```
server IP_address_of_time_provider2_within_same_network
```

or

To use auto configuration, add lines similar to the following to each time consumer's `ntp.conf` file:

```
broadcastclient subnet_broadcast_address
```

or

```
multicastclient
```

4.3.4 Sample Scenario

The sample scenario in the following figure demonstrates how to configure a setup using wide area configuration.

Figure 4-5   *Sample Scenario for Wide Area Configuration*
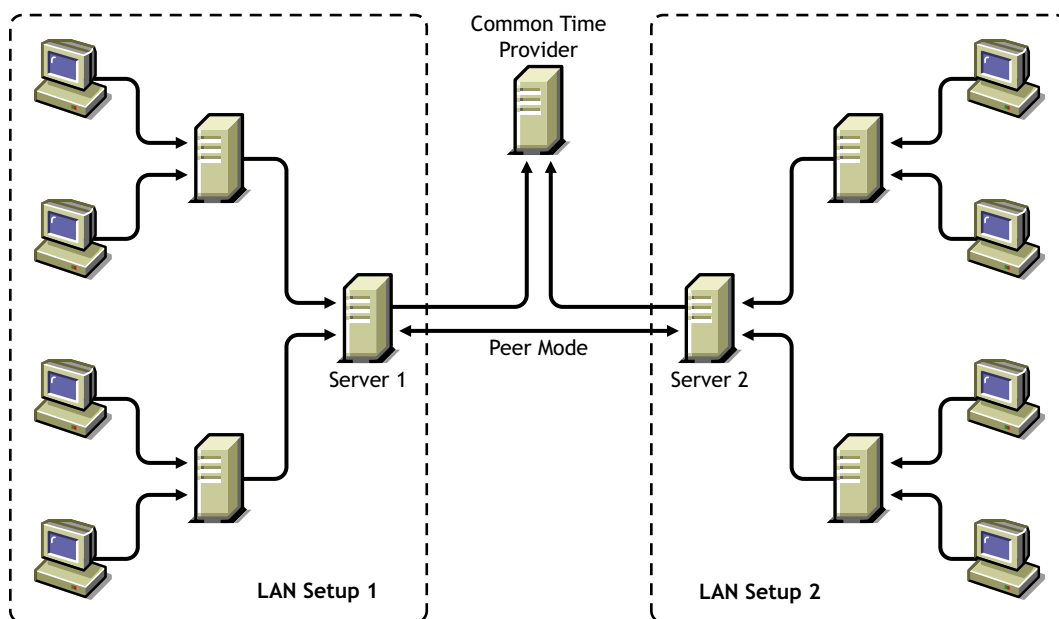
In this scenario:

- There are two LAN setups. The time consumers and time providers in both the setups are synchronized for time.

  Refer to Section 4.1, "Manual Configuration," on page 19 and Section 4.2, "Auto Configuration," on page 22 for more information.

- In both LAN setups, only Server1 and Server2 obtain time from sources outside the local network.

- Server1 and Server2 are time consumers that do the following:

  - First, obtain time from a common time provider in the client-server mode.

  - Second, obtain time from each other in the peer mode.

  - Third, fall back to their local clock if the external time sources fail.

To use auto configuration to configure the setup in this scenario:

1 Configure Server1 to obtain time from the common time provider in the client-server mode and make this the preferred time provider by adding a line similar to the following to Server1's `ntp.conf` file:

   `server IP_address_of_Common_Time_Provider prefer`

2 Configure Server1 to obtain time from its peer, Server2 in the peer mode.

   `peer IP_address_of_Server2`

3 Configure Server1 to fall back to its own local clock. Set a low stratum value for the local clock so that preference is given to the external time sources. By setting a low stratum value, it falls back to its local clock only when the other sources fail.

   `server 127.127.1.0`

   `fudge 127.127.1.0 stratum 3`

4 Load XNTPD for the changes to take effect by entering the following at command prompt:

   `Load XNTPD`

5 Repeat this procedure, with the appropriate changes to Server2's `ntp.conf` file, to configure Server2.

You are minimizing network traffic by configuring only Server1 and Server2 to obtain time from a time provider outside the local network.

4.4 Synopsis

The following table summarizes which lines should to be added to the `ntp.conf` file in each scenario. This file is located in `sys:\etc\ntp.conf`.

Table 4-2 *Synopsis*

Configuration Type	Time Provider	Time Consumer	Best-Suited Scenario
Manual configuration for client-server mode	`server IP_address_of_time _provider prefer`	`server IP_address_of_time _provider prefer`	◆ Fewer than 15 servers ◆ In the same geographical location

Manual configuration for self-synchronization	`server 127.127.1.0 prefer`
	
	`fudge 127.127.1.0 stratum 3`
		N/A	N/A

Auto configuration for broadcast	`broadcast subnet_broadcast_a ddress key key_ID`	`broadcastclient subnet_broadcast_a ddress`	◆ Time provider changes frequently ◆ Time provider is unknown
Auto configuration for multicast	`broadcast 224.0.1.1 key key_ID`	`multicastclient`	◆ Time provider changes frequently ◆ Time provider is unknown ◆ Generates less traffic when compared to broadcast

Wide area configuration for peer mode or a combination of more than one mode with fault tolerance	`server IP_address_of_exte rnal_time_source`
	
	`peer IP_address_of_peer`
	
	`server 127.127.1.0`
	
	`fudge 127.127.1.0 stratum 3`
		Either manual or auto type can be used:
		
		`server IP_address_of_time _provider1_within_ same_network`
		
		`server IP_address_of_time _provider_2_within _same_network`
		
		or
		
		`broadcastclient subnet_broadcast_a ddress`
			◆ Servers span across geographical locations.
			
			◆ Servers are synchronized in a combination of more than one mode.
			
			◆ Need for fault tolerance and minimizing network traffic.

4.5 Remote Configuration

NTP can be configured remotely from Novell® Remote Manager .

Novell Remote Manager displays the configuration file (`ntp.conf`) of all the servers in the eDirectory™ 8.7.3 or later, tree that you have authenticated to.

You have the following options after making changes to the `ntp.conf` file:

- **Save:** Saves the configuration file.
- **Apply:** Saves the configuration file and restarts XNTPD to reflect the new changes.

  NOTE: If a tree is configured with multiple servers running NTP and Timesync, clicking *Apply* for the server running Timesync unloads Timesync and loads XNTPD.

- **Restart:** Restarts XNTPD without saving the changes.

Migrating Timesync Servers to NTP

5

You can now use the iManager Web-based administration tool to migrate servers from Timesync to NTP.

To migrate the servers, perform the following tasks in the order listed below:

- Section 5.1, "Reviewing Pre-requisites," on page 31
- Section 5.2, "Selecting the Mode of Migration," on page 31
- Section 5.3, "Configuring Servers Running the NTP Services," on page 34
- Section 5.4, "Monitoring the Health Status of Migrated Servers," on page 35

5.1 Reviewing Pre-requisites

☐ Ensure that the NTP iManager plug-in (`ntptimesync.npm`) is located in the root directory of the product. For more information see "Downloading and Installing the iManager Plug-in" in the *Novell iManager 2.7.3 Administration Guide*.

☐ The servers to be migrated must be Netware® 6.5 servers existing on the same Novell® eDirectory™ 8.7.3 or later tree and must be running on Timesync.

5.2 Selecting the Mode of Migration

1 In iManager, click *Time Synchronization > Migration* to display the Migrate Time Servers page.
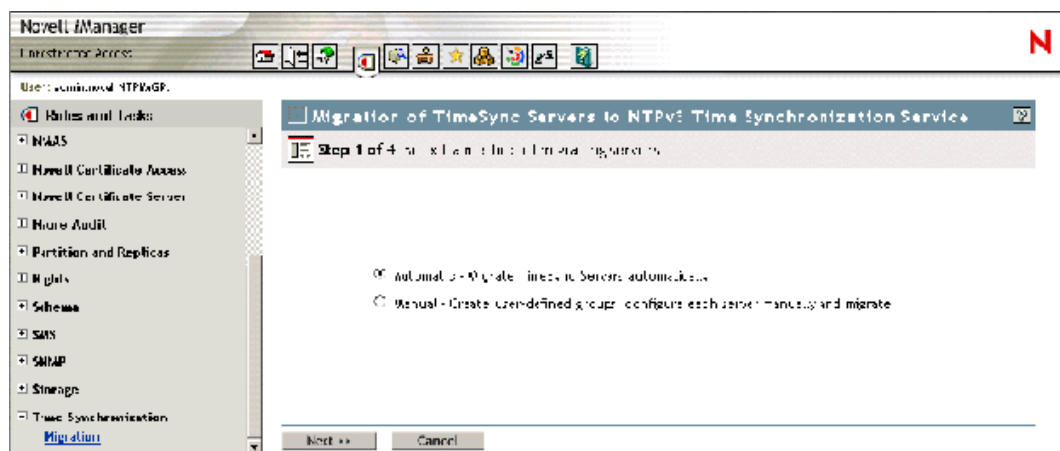
2 Select the method of migration, *Automatic* or *Manual*.

- Section 5.2.1, "Automatically Migrating Servers," on page 32
- Section 5.2.2, "Manually Migrating Servers," on page 32

5.2.1 Automatically Migrating Servers

This is the default migration method and it can be used by a user who is new to NTP and prefers time synchronization services to be automatically set up in the eDirectory tree. The servers are migrated to NTPv3 based on a one-to-one mapping of `timesync.cfg` to `ntp.conf`.

1 Select the servers to be migrated.
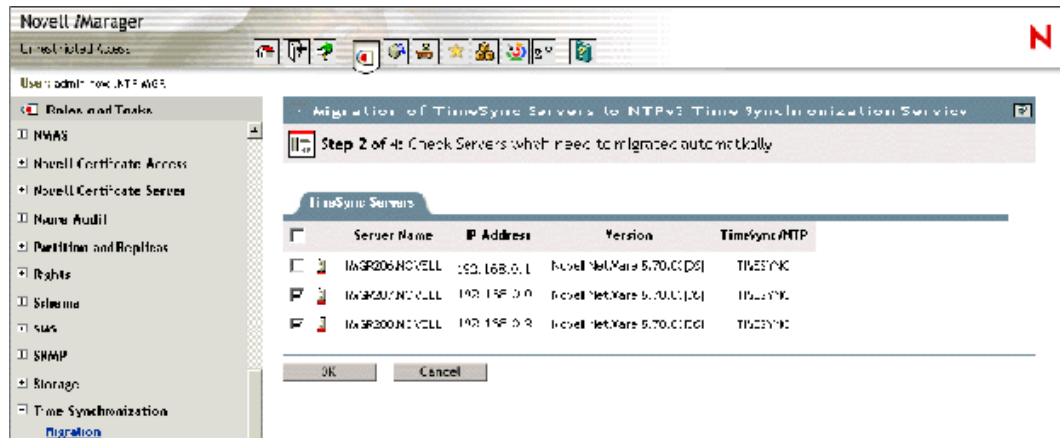
2 Click *OK* to confirm the migration.

A time-out screen is displayed, followed by the status of monitored servers.

For more information on monitoring the status of migrated servers, see Section 5.4, "Monitoring the Health Status of Migrated Servers," on page 35.

5.2.2 Manually Migrating Servers

This method can be used by a user who is familiar with NTP. It helps you create a group of servers and configure individual servers. Do the following tasks in order:

- "Adding or Removing Timesync Groups" on page 32
- "Adding or Removing Servers from a Group" on page 33

Adding or Removing Timesync Groups

To add a Timesync Group:

1 On the Timesync Server Group page, click *Add Group*.
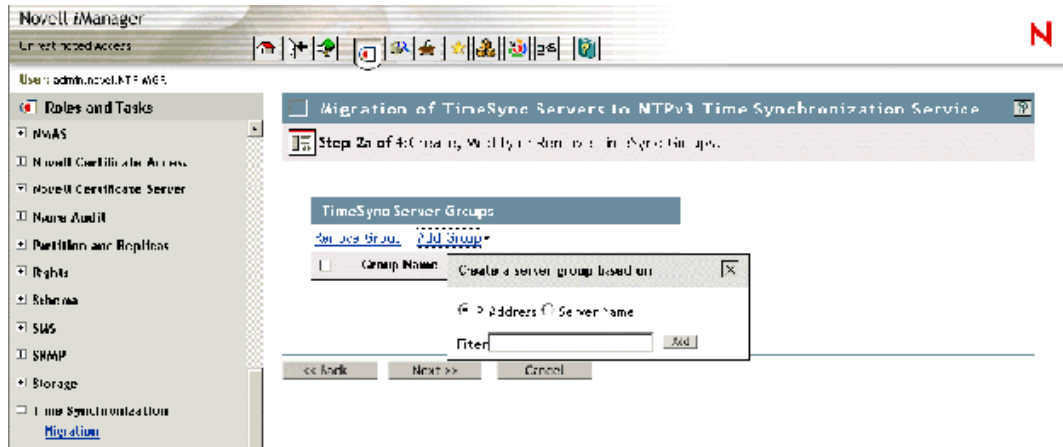
2 Select the servers to be added to the group, based on the IP address or server name filter, then click *Next*.

IMPORTANT: The only permitted wildcard filter is the asterisk (*)

To remove a Timesync Group:

On the Timesync Server Group page, select the group to be excluded and click *Remove Group*.

Adding or Removing Servers from a Group

To add a server to a Group:

1 Click *Add Server* to display a drop-down list.
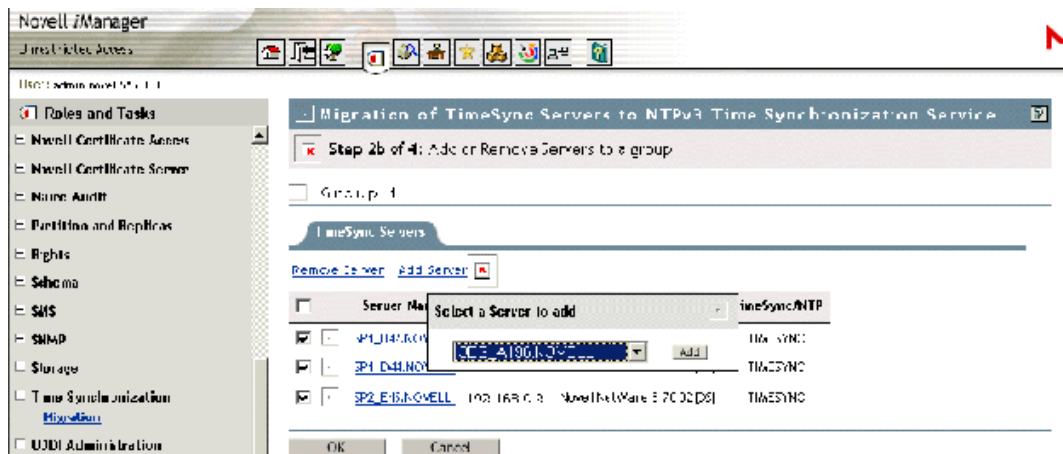
2 Select the server you want, then click *Add*.

To remove a server from Group:

1 Select one or more servers from the list displayed, then click *Remove Server*.

5.3 Configuring Servers Running the NTP Services

In an automatic migration, all the configuration for migrating the servers is done through backend process.

During a manual migration, you can configure the parameters of each server in a group through a simple or advanced configuration process.

- Section 5.3.1, "Simple Configuration," on page 34
- Section 5.3.2, "Advanced Configuration," on page 35

5.3.1 Simple Configuration

Each server in a particular group can be configured individually on certain selected NTP parameters.

1 On the Configuration Parameters page, specify values for *Time Provider* and *Peer*.

2 Specify the options to enable on the time provider.
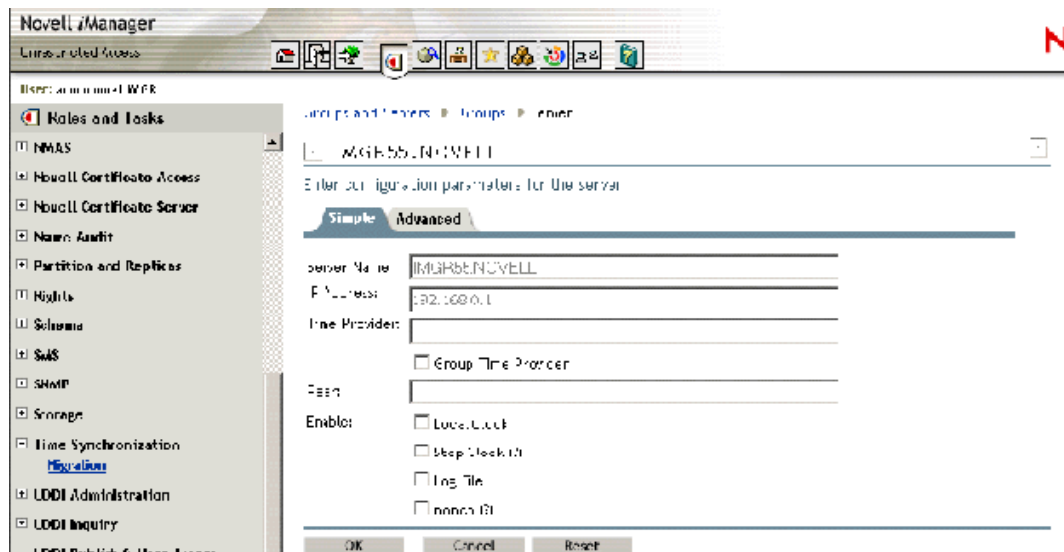
3 Click *OK* to update the configuration file of the server.

4 Click *Reset* to apply the changes made.

5.3.2 Advanced Configuration

This mode of editing is recommended for users familiar with NTP. It helps you to edit the `ntp.conf` file for making configuration changes.

1 After configuring the server parameters, Click *OK* to update them.
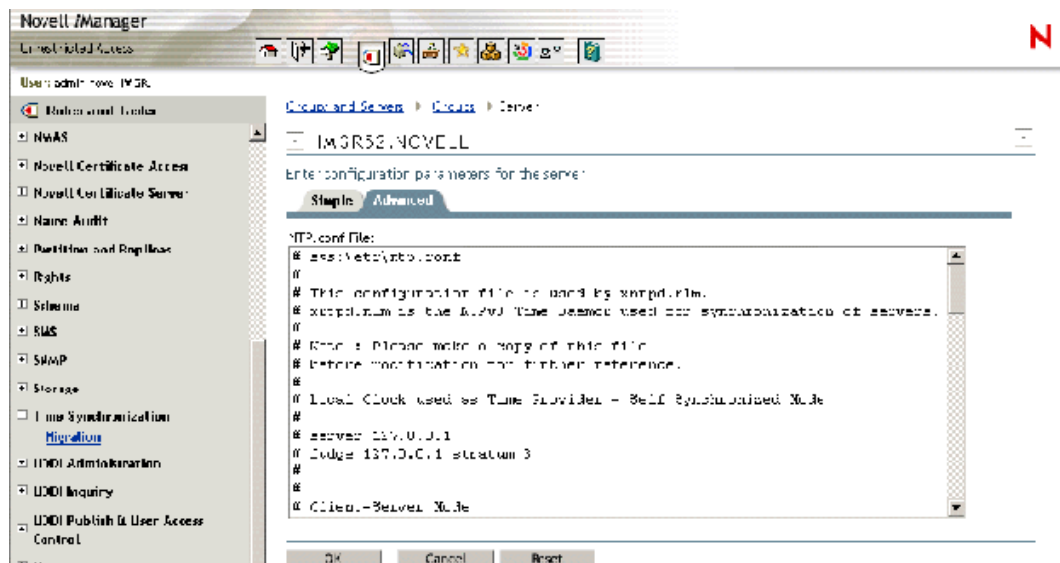
5.4 Monitoring the Health Status of Migrated Servers

The Monitor Configured Groups page lets you view the list of servers which have been migrated.

The successful servers are the ones which are correctly configured and for which the configurations are properly updated. The unsuccessful servers for each group consist of servers for which the configurations are not applied for the particular remote server. Continue from previous section and use the following step:

1 Click *OK* to end the migration task.
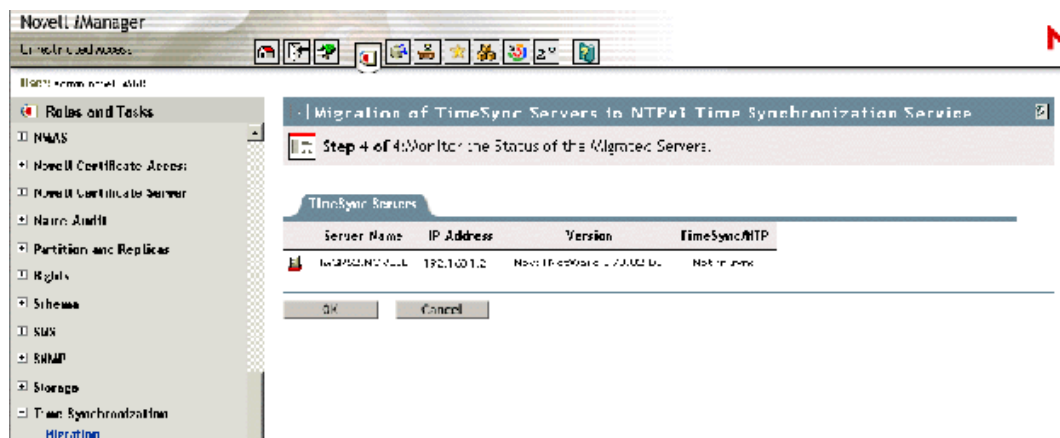
This page enables you to detect whether `xntpd.nlm` is successfully loaded or not.

NTP Utilities

<div style="text-align: right; font-size: large;">6</div>

This section explains the following NTP utilities necessary to synchronize time:

6.1 NTPDate

NTPDate is used to set the local time and date by using NTP servers. The NTP servers are determined by specifying them in the NTPDate command line argument.

Usage:

```
NTPDate [-bBdhoquv] [-a key] [-e authdelay] [-k keyfile] [-o version] [-p
samples] [-s logfile] [-t timeout] server [ ... ]
```

Table 6-1 *NTPDate Parameters*

Parameter	Description
-a *key*	Enables the authentication function and specifies the key identifier to be used for authentication as the argument keyntpdate. The keys and key identifiers must match in both the client and server key files.
	The default is to disable the authentication function.
-B	Forces the time to always be slewed, even if the measured offset is greater than + or - 128 ms.
	The default is to step the time if the offset is greater than + or - 128 ms.
	NOTE: If the offset is greater than + or - 128 ms, it takes a long time (hours) to slew the clock to the correct value. During this time, the host should not be used to synchronize clients.
-b	Forces the time to be stepped, rather than slewed (default). This option should be used when called from a startup file at boot time.
-d	Enables the debugging mode. In the debugging mode, NTPDate goes through all the steps, but does not adjust the local clock. Information useful for general debugging is also displayed.
-e *authdelay*	Specifies the processing delay to perform an authentication function as the value authdelay, in seconds and fraction (see Section 6.5, "XNTPD," on page 44 for details). This number is usually small enough to be negligible for most purposes, although specifying a value might improve timekeeping on very slow CPUs.
-h	Displays the help.

Parameter	Description
-k *keyfile*	Specifies the path for the authentication key file as the string keyfile. The default is `sys:\etc\ntp.keys`. This file should be in the format described in XNTPD.
-o *version*	Specifies the NTP version for outgoing packets as the integer version, which can be 1 or 2. This allows NTPDate to be used with older NTP versions. **IMPORTANT:** If the -o option is not used, the default is set to 3.
-p *samples*	Specifies the number of samples to be acquired from each server as the integer samples, with values from 1 to 8 inclusive. The default is 4.
-q	Query only. Do not set the clock.
-s *logfile*	Enables logging of the XNTPD progress screen into the *logfile*.
-t *timeout*	Specifies the maximum time waiting for a server response as the value timeout, in seconds and fractions. The value is rounded off to a multiple of 0.2 seconds. The default is 1 second, which is a value suitable for polling across a LAN.
-u	Directs NTPDate to use an unprivileged port or outgoing packets. This is most useful when you are behind a firewall that blocks incoming traffic to privileged ports and you want to synchronize with hosts beyond the firewall. The -d option always uses unprivileged ports.
-v	Be verbose. This option logs NTPDate's version identification string.

6.2 NTPQ

NTPQ is an interactive NTPv3 query utility to help query the status or quality of time parameters.

Usage:

```
NTPQ [-i np] [-c command] [host] [ ... ]
```

Table 6-2 *NTPQ Parameters*

Parameter	Description
-c	Is interpreted as an interactive format command and is added to the list of commands to be executed on the specified hosts. Multiple -c options can be given.
-i	Forces NTPQ to operate in the interactive mode. Prompts are written to the standard output and commands read from the standard input.
-n	Output all host addresses in dotted-quad numeric format rather than converting to the canonical hostnames.
-p	Displays a list of the peers known to the server as well as a summary of their state.

6.2.1 Internal Commands

Interactive format commands consist of a keyword followed by zero to four arguments. Only enough characters of the full keyword to uniquely identify the command need be typed. The output of a command is normally sent to the standard output, but optionally the output of individual commands can be sent to a file by appending a "<" followed by a filename, to the command line. A number of interactive format commands are executed entirely within the NTPQ program itself and do not result in NTP mode 6 requests being sent to a server. These are described in the following table.

Table 6-3 *Parameter Description*

Parameter	Description
? [*command_keyword*] helpl [*command_keyword*]	A "?" by itself displays a list of all the command keywords known to this incarnation of NTPQ. A "?" followed by a command keyword displays the function and usage information about the command.
addvars *variable_name* [= *value*] [...] rmvars *variable_name* [...] clearvars	The data carried by NTP mode 6 messages consists of a list of items of the form variable_name = value, where the " = value" is ignored, and can be omitted in requests to the server to read variables. NTPQ maintains an internal list in which data to be included in control messages can be assembled and sent using the readlist and writelist commands described below. The addvars command allows variables and their optional values to be added to the list. If more than one variable is to be added, the list should be comma-separated and not contain white space. The rmvars command can be used to remove individual variables from the list, and the clearlist command removes all variables from the list.
authenticate yes | no	Normally NTPQ does not authenticate requests unless they are write requests. The authenticate *yes* command causes NTPQ to send authentication with all requests it makes. Authenticated requests causes some servers to handle requests slightly differently, and the CPU is caught CPU in fuzzball (Fuzzballs are clocks that survey and average ticks from many different clocks and thereby reliable) server cycles if you turn authentication on before doing a peer display.
cooked	Causes output from query commands to be "cooked," which means that variables that are recognized by the server have their values reformatted for human consumption. Variables that NTPQ determines should have a decodeable value but didn't are marked with a trailing "?".
debug more | less | off	Turns internal query program debugging on and off.
delay *milliseconds*	Specify a time interval to be added to time stamps included in requests that require authentication. This is used to enable (unreliable) server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. However, the server does not now require time stamps in authenticated requests, so this command might be obsolete.
host *hostname*	Set the host to which future queries are sent. Hostname can be either a name or a numeric address.

Parameter	Description
hostnames [yes | no]	If Yes is specified, hostnames are printed in information displays. If No is specified, numeric addresses are displayed instead. The default is Yes, unless modified using the command line *-n* switch.
keyid *keyid*	This command allows the specification of a key number to be used to authenticate configuration requests. This must correspond to a key number the server has been configured to use for this purpose.
ntpversion 1 | 2 | 3	Sets the NTP version number that NTPQ claims in packets. Defaults to 3, Note that mode 6 control messages (and modes, for that matter) didn't exist in NTP version 1. There appear to be no servers left that demand version 1.
quit	Exit NTPQ.
passwd	This command prompts you to type a password (which is not echoed) which is used to authenticate configuration requests. The password must correspond to the key configured for use by the NTP server for this purpose if such requests are to be successful.
raw	Causes all output from query commands to be displayed as received from the remote server. The only formatting/intepretation done on the data is to transform non-ASCII data into a printable form.
timeout *millseconds*	Specifies a timeout period for responses to server queries. The default is about 5000 milliseconds. Because NTPQ retries each query once after a timeout, the total waiting time for a timeout is twice the timeout value set.

6.2.2 Control Message Commands

Each peer known to an NTP server has a 16-bit integer association identifier assigned to it. NTP control messages that carry peer variables must identify the peer that the values correspond to by including its association ID. An association ID of 0 is special, and indicates that the variables are system variables, whose names are drawn from a separate name space.

Control message commands result in one or more NTP mode 6 messages being sent to the server, and cause the data returned to be printed in some format. Most commands currently implemented send a single message and expect a single response. The current exceptions are the `peers` command, which sends a preprogrammed series of messages to obtain the data it needs, and the `mreadlist` and `mreadvar` commands, which iterate over a range of associations.

Table 6-4 *Parameter Description*

Parameter	Description
associations	Obtains and displays a list of association identifiers and peer statuses for in-spec peers of the server being queried. The list is displayed in columns. The first of these is an index numbering the associations from 1 for internal use, the second is the actual association identifier returned by the server and the third is the status word for the peer. This is followed by a number of columns containing data decoded from the status word. The data returned by the associations command is cached internally in NTPQ. The index is then of use when dealing with servers that use association identifiers that are hard for humans to type, because for any subsequent commands that require an association identifier as an argument, the form and index can be used as an alternative.
clockvar [*assocID*] [*variable_name* [= cv [*assocID*] [*variable_name* [= *value* [...]] [...]	Requests that a list of the server's clock variables be sent. Servers that have a radio clock or other external synchronization respond positively to this. If the association identifier is omitted or has a value of zero, the request is for the variables of the system clock and generally gets a positive response from all servers with a clock. If the server treats clocks as pseudo-peers, so more than one clock can be connected at once, referencing the appropriate peer association ID shows the variables of a particular clock period. Omitting the variable list causes the server to return a default variable display.
lassocations	Obtains and displays a list of association identifiers and peer statuses for all associations for which the server is maintaining state. This command differs from the associations command only for servers that retain state for out-of-spec client associations (such as fuzzballs). Such associations are normally omitted from the display when the associations command is used, but are included in the output of lassociations.
lpassociations	Displays data for all associations, including out-of-spec client associations, from the internally cached list of associations. This command differs from passociations only when dealing with fuzzball servers.
lpeers	Like R peers, except a summary of all associations for which the server is maintaining state is displayed. This can produce a much longer list of peers than fuzzball servers.
mreadlist *assocID assocID* mrl *assocID assocID*	Like the readlist command, except the query is done for each of a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent associations command.
mreadvar *assocID assocID* [*variable_name* [= *value* [...] mrv *assocID assocID* [*variable_name* [= *value* [...]	Like the readvar command, except the query is done for each of a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent associations command.
opeers	An old form of the peers command with the reference ID replaced by the local interface address.

Parameter	Description
passociations	Displays association data concerning in-spec peers from the internally cached list of associations. This command performs identically to the `associations` command except that it displays the internally stored data rather than making a new query.
peers	
pstatus *assocID*	Sends a read status request to the server for the given association. The names and values of the peer variables returned are printed. The status word from the header is displayed preceding the variables, both in hexadecimal and in simple english.
readlist [*assocID*] rl [*assocID*]	Requests that the values of the variables in the internal variable list be returned by the server. If the association ID is omitted or is 0, the variables are assumed to be system variables. Otherwise they are treated as peer variables. If the internal variable list is empty, a request is sent without data, which should induce the remote server to return a default display.
readvar *assocID variable_name* [= *value*] [...] rv *assocID* [*variable_name* [= *value*] [...]	Requests that the values of the specified variables be returned by the server by sending a read variables request. If the association ID is omitted or is given as zero, the variables are system variables; otherwise, they are peer variables and the values returned are those of the corresponding peer. Omitting the variable list sends a request with no data that should induce the server to return a default display.
rvi *index*	Similar to rv with an association ID corresponding to this index.
writevar *assocID variable_name* [= *value*] [...]	Similar to readvar request, except the specified variables are written instead of read.
writelist [*assocID*]	Similar to readlist request, except the internal list variables are written instead of read.
showipconf	Displays the IP addresses of the local machine along with the corresponding subnet broadcast addresses.

6.3 NTPTrace

NTPTrace is a utility to query the time server and its servers until the master server is queried. NTPTrace determines where a given NTP server gets its time from, and follows the chain of NTP servers back to their master time source. If given no arguments, it starts with local host. An example of the output from NTPTrace is given below:

```
% ntptrace

localhost: stratum 4, offset 0.0019529, synch distance 0.144135

server2ozo.com: stratum 2, offset 0.0124263, synch distance 0.115784

usndh.edu: stratum 1, offset 0.0019298, synch distance 0.011993, refid

'WWVB'
```

On each line, the fields are (left to right):

- ◆ hostname

- host stratum; stratum is the server hop count to the primary source.
- time offset between that host and the local host (as measured by NTPTrace; this is why it is not always zero for "local host"). The time unit is given in seconds.
- host synchronization distance; synchronization distance is the estimated error relative to the primary source
- reference clock ID (only for stratum-1 servers)

Usage:

```
NTPTrace [ -dhnv ] [ -r retries ] [ -t timeout ] [ server ]
```

Table 6-5 *NTPTrace Parameters*

Parameter	Description
-d	Turns on debugging output.
-h	Displays the help.
-n	Turns off the printing of hostnames; instead, host IP addresses are given. This can be useful if a nameserver is down.
-r *retries*	Sets the number of retransmission attempts for each host (default = 5).
-t *timeout*	Sets the retransmission timeout (in seconds) (default = 2).
-v	Displays verbose information about the NTP servers.

6.4 XNTPDC

XNTPDC is the remote configuration utility. It is used to query the XNTPD daemon about its current state and to request changes in that state.

If one or more request options are included on the command line when XNTPDC is executed, each of the requests is sent to the NTP servers running on each of the hosts given as command line arguments, or on localhost by default. If no request options are given, XNTPDC attempts to read commands from the standard input and executes these on the NTP server running on the first host given on the command line, again defaulting to localhost when no other host is specified. XNTPDC prompts for commands if the standard input is a terminal device.

The operations of XNTPDC are specific to the particular implementation of the XNTPDC daemon and can be expected to work only with that implementation, and possibly some previous versions of the daemon. Requests from a remote XNTPDC program that affect the state of the local server must be authenticated, which requires both the remote program and local server to share a common key and key identifier.

```
XNTPDC [ -i lnps ] [ -c command ] [ host ] [ ... ]
```

Specifying a command line option other than -i or -n causes the specified query (queries) to be sent to the indicated hosts immediately. Otherwise, XNTPDC attempts to read interactive format commands from the standard input.

Table 6-6 *XNTPDC Parameters*

Parameter	Description
-c *command*	The following argument is interpreted as an interactive format command and is added to the list of commands to be executed on the specified hosts. Multiple -c options can be given.
-i	Forces XNTPDC to operate in interactive mode. Prompts are written to the standard output and commands are read from the standard input.
-l	Obtains a list of peers known to the servers. This switch is equivalent to -c listpeers.
-n	Outputs all host addresses in dotted-quad numeric format rather than converting to the canonical hostnames.
-p	Displays a list of the peers known to the server as well as a summary of their state. This is equivalent to -c peers.
-s	Displays a list of the peers known to the server as well as a summary of their state, but in a slightly different format than the -p switch. This is equivalent to -c dmpeers.

6.5 XNTPD

XNTPD is an operating system daemon that sets and maintains the system time of day in synchronization with Internet standard time servers.

The daemon can operate in any of several modes, including client-server and broadcast/multicast mode, as described in RFC-1305. A broadcast/multicast client can discover remote servers, compute client-server propagation delay correction factors, and configure itself automatically. This makes it possible to deploy numerous workstations without specifying configuration details specific to the local environment.

Ordinarily, XNTPD reads the ntp.conf configuration file at startup in order to determine the synchronization sources and operating modes. It is also possible to specify a working, although limited, configuration entirely on the command line, obviating the need for a configuration file. This might be particularly appropriate when the local host is to be configured as a broadcast or multicast client, with all peers being determined by listening to broadcasts at run time.

Various internal XNTPD variables can be displayed and configuration options altered while the daemon is running using the NTPQ and XNTPDC utility programs.

Usage:

```
XNTPD [ -aAbdhm ] [ -c configfile ] [ -f driftfile ] [ -k keyfile ] [ -l
logfile ] [-n log file limit] [ -p pidfile ] [ -r broadcastdelay ] [ -s
statsdir ] [ -t trustkey ] [ -v variable ] [ -V defaultvariable ] [-T noncp/slp
] [-S]
```

Table 6-7 *XNTPD Parameters*

Parameter	Description
-a	Enables authentication mode. The default is enabled, so this option is obsolete now.
-A	Disables authentication mode.
-b	Synchronizes by using NTP broadcast messages.
-c *configfile*	Specifies the name and path of the configuration file.
	NOTE: Novell® Remote Manager does not understand this user-defined configuration file, so it opens the default `sys:\etc\ntp.conf` file.
-d	Specifies the debugging mode. This flag might occur multiple times, with each occurrence indicating greater detail of display.
-f *driftfile*	Specifies the name and path of the drift file.
-h	Displays the help.
-k *keyfield*	Specifies the name and path of the file containing the NTP authentication keys.
-l *logfile*	Specifies the name and path of the log file. The default is the system log facility.
	NOTE: If the -S option (see for more information) is enabled along with -l option, the NTPDate events are also logged into the log file (`ntpdate.log`).
-n	Indicates the log file limit.
-m	Synchronizes by using NTP multicast messages on the IP multicast group address 224.0.1.1 (requires multicast kernel).
-p *pidfile*	Specifies the name and path to record the daemon's process ID.
-r *broadcastdelay*	Specifies the default propagation delay from the broadcast/multicast server and this computer. This is used only if the delay cannot be computed automatically by the protocol.
-s *statsdir*	Specifies the directory path for files created by the statistics facility.
-t *key*	Adds a key number to the trusted key list.
-v *variable*	Adds a system variable.
-V *defaultvariable*	Adds a system variable listed by default.
-T *noncp*	Provides Timesync migration or backward compatibility options.
	Prevents running of the NCP™ engine on XNTPD, which services all NCP time requests from NetWare® 4, Novell clients, and dsrepair.
-T *slp*	Enables NTP to automatically discover SLP advertising a Timesync SINGLE server on the network and add the Timesync SINGLE server's IP address in the `ntp.conf` configuration file as a time provider.
	WARNING: Do not use this option in the `sys:\system\timeserv.ncf` file.

Parameter	Description
-S	XNTPD steps the clock to the time of the best available server by calling NTPDate with the server list from the NTP configuration file (ntp.conf). This sets the clock status to "nearly in sync", meaning that time is synchronized in as close as 0.5 seconds. This basically helps XNTPD to synchronize† quickly.

6.5.1 The Configuration File

The XNTPD configuration file (`ntp.conf`) is read at initial startup in order to specify the synchronization sources, modes and other related information. It is installed in the `sys:\system` directory, but could be installed elsewhere (see "-c configfile" on page 45).

The `ntp.conf` looks similar to the following:

```
# sys:\etc\ntp.conf

#

# This configuration file is used by xntpd.nlm.

# xntpd.nlm is the NTPv3 Time Daemon used for

# synchronization of servers.

#

# Note : Please make a copy of

#    this file before modification

#    for further reference.

#

# Local Clock used as Time Provider - Self Synchronized Mode

#

# server 127.127.1.0

# fudge 127.127.1.0 stratum 3

#

# Client-Server Mode

# <IP Address> : Time provider IP address
```

```
#
# Time Provider
# server <IP Address>
#
# Time Provider
# server <IP Address>
#
# Peer-Peer Mode
# <IP Address> : Peer IP address
#
# peer <IP Address>
#
# To Configure this NetWare box to Broadcast the "time service"
#
# broadcast <Subnet broadcast Address> key <key_id>
#               or
# broadcast 255.255.255.255 key <key_id>
#
# To Configure this NetWare box to Multicast the "time service"
#
# broadcast 224.0.1.1 key <key_id>
#
# To Configure NTP Broadcast Client
#
# broadcastclient
#
# To Configure NTP Multicast Client
#
# multicastclient
#
# Authentication Options
#
# enable auth monitor
```

```
# keys sys:\etc\ntp.keys

# trustedkey 0

# requestkey 0

# controlkey 0

#

# Backward Compatibility with Timesync

#

# Switch off the Timesync NCP service

# noncp

# Step the time to the source clock for slewing

# stepclock

#

# Monitoring/Logging Options

#

  driftfile sys:\system\drift.ntp

# statsdir sys:\system\

# logfile sys:\system\ntp.log

# filegen peerstats file peerstat type none enable

# filegen loopstats file loopstat type none enable

# filegen clockstats file clkstat type none enable
```

Configuration commands consist of an initial keyword followed by a list of arguments, some of which can be optional, separated by white space. Commands cannot be continued over multiple lines. Arguments can be host names, host addresses written in numeric, dotted-quad form, integers, floating point numbers (when specifying times in seconds) and text strings. Optional arguments are delimited by [ ] in the following descriptions, while alternatives are separated by |. The notation [...] means an optional, indefinite repetition of the last item before the [...].

See the following for configuration and control options. Although there is a rich set of options available, the only required option is one or more server, peer, or broadcast commands described in "Configuration Options" on page 49.

6.5.2 Files

`sys:\etc\ntp.conf`. The default name of the configuration file.

`sys:\system\ntp.drift`. The default name of the drift file.

`sys:\etc\ntp.keys`. The default name of the key file.

6.5.3 Configuration Options

```
peer address [ key key ] [ version version ] [ prefer ] [ minpoll minpoll [
maxpoll maxpoll ]
```

```
server address [ key key ] [ version version ] [ prefer ]
```

```
broadcast address [ key key ] [ version version ] [ ttl ttl ]
```

These three commands specify the time server name or address to be used and the mode in which to operate. The address can be either a DNS name or a IP address in dotted-quad notation. The `peer` command specifies that the local server is to operate in symmetric active mode with the remote server. In this mode, the local server can be synchronized to the remote server and, in addition, the remote server can be synchronized by the local server. This is useful in a network of servers where, depending on various failure scenarios, either the local or remote server might be the better source of time.

The `server` command specifies that the local server is to operate in client mode with the specified remote server. In this mode, the local server can be synchronized to the remote server, but the remote server can never be synchronized to the local server.

The `broadcast` command specifies that the local server is to operate in broadcast mode, where the local server sends periodic broadcast messages to a client population at the broadcast/multicast address specified. Ordinarily, this specification applies only to the local server operating as a sender; for operation as a broadcast client, see the `broadcastclient` or `multicastclient` commands below. In this mode, address is usually the broadcast address on (one of) the local networks or a multicast address assigned to NTP. The IANA organization has assigned the address 224.0.1.1 to NTP; this is presently the only address that should be used.

NOTE: The use of multicast features requires a multicast kernel, which is not yet ubiquitous in vendor products.

For more information on the configuration options, see Table 6-8.

Table 6-8 *XNTPD Configuration Options*

Parameter	Description
key *key*	All packets sent to the address are to include authentication fields encrypted by using the specified key identifier, which is an unsigned 32-bit integer. The default is to not include an encryption field.
version *version*	Specifies the version number to be used for outgoing NTP packets. Versions 1, 2, and 3 are the choices, with version 3 the default.

Parameter	Description
prefer	Marks the server as preferred. All other things being equal, this host is chosen for synchronization among a set of correctly operating hosts.
ttl *ttl*	This option is used only with broadcast mode. It specifies the time- to-live ttl to use on multicast packets. The default is 127.
minpoll *minpoll*	This option specifies the minimum polling interval for NTP messages, in seconds to the power of two. The allowable range is 4 (16 s to 14 (16384 s) inclusive. The default is 6 (64 s) for all except reference clocks.
maxpoll *maxpoll*	This option specifies the maximum polling interval for NTP messages, in seconds to the power of two. The allowable range is 4 (16 s to 14 (16384 s) inclusive. The default is 10 (1024 s) for all except reference clocks.
broadcastclient [*address*]	This command directs the local server to listen for broadcast messages at the broadcast address address of the local network. The default address is the subnet address with the host field bits set to ones. Upon hearing a broadcast message for the first time, the local server measures the nominal network delay by using a brief client/server exchange with the remote server, then enters the broadcastclient mode, in which it listens for and synchronizes to succeeding broadcast messages. In order to avoid accidental or malicious disruption in this mode, both the local and remote servers should operate by using authentication and the same trusted key and key identifier.
multicastclient [*address*] [...]	This command directs the local server to listen for multicast messages at the group addresses of the global network. The default address is assigned by the IANA organization to NTP (224.0.1.1). This command operates in the same way as the `broadcastclient` command, but uses IP multicasting. Support for this command requires a multicast kernel.
driftfile *driftfile*	This command specifies the name of the file used to record the frequency offset of the local clock oscillator. If the file exists, it is read at startup in order to set the initial frequency offset and then updated once per hour with the current frequency offset computed by the daemon. If the file does not exist or this command is not given, the initial frequency offset is assumed to be zero. In this case, it might take several hours for the frequency to stabilize and the residual timing errors to subside.

The `ntp.drift` file format consists of a single line containing a single floating point number, which records the frequency offset measured in parts-per-million (PPM). That the file is updated once per hour by first writing the current drift value into a temporary file and then renaming this file to replace the old version. This implies that XNTPD must have write permission for the directory the drift file is located in, and that file system links, symbolic or otherwise, should probably be avoided. |

```
enable auth | bclient | monitor | pll | pps | stats
```

```
disable auth | bclient | monitor | pll | pps | stats
```

Provides a way to enable or disable various server options. Flags not mentioned are unaffected.

NOTE: All these flags can be controlled remotely by using XNTPDC.

Table 6-9 *XNTPD Parameters for Enabling and Disabling Server Options*

Parameter	Description
auth	Enables the server to synchronize with unconfigured peers only if the peer has been correctly authenticated by using a trusted key and key identifier. The default for this flag is enable.
bclient	Enables the server to listen for a message from a broadcast or multicast server, as the `multicastclient` command does with the default address. The default for this flag is disable.
monitor	Enables the monitoring facility. See the XNTPDC program and the `monlist` command or further information. The default for this flag is enable.
pll	Enables the server to adjust its local clock by means of NTP. If disabled, the local clock free-runs at its intrinsic time and frequency offset. This flag is useful in case if the local clock is controlled by some other device or protocol and NTP is used only to provide synchronization to other clients. In this case, the local clock driver is used. The default for this flag is enable.
pps	Enables the pulse-per-second (PPS) signal when frequency and time is disciplined by the precision time kernel modifications. The default for this flag is disable.
stats	Enables the statistics facility. The default for this flag is enable.
tick *value*	If no value for tick can be found from the kernel, use this value. This is the "normalized" value; if your system keeps tick in nanoseconds you must divide your value by 1000. The expected range of the value is between 900 and 11,000 (don't use the comma in the config file).
tickadj *value*	If no value for tickadj can be found in the kernel, use this value. The value must be "normalized"; if your system keeps tickadj in nanoseconds you must divide your value by 1000. The expected range of the value is between 1 and 50.

The XNTPD `-S` and `-T noncp` options can also be added in the configuration file as `stepclock` and `noncp` respectively.

Table 6-10 *Stepclock and Noncp*

Parameter	Description
stepclock	XNTPD steps the clock to the time of the best available server by calling NTPDate with the server list from the NTP configuration file (`ntp.conf`). This sets the clock status to "nearly in sync", meaning that time is synchronized as close as 0.5 seconds. This helps XNTPD to synchronize quickly.
noncp	Prevents running of the NCP engine on XNTPD, which services all NCP time requests from NetWare 4, Novell clients, and dsrepair.

6.5.4 Authentication Options

The NTP standard specifies an extension that provides cryptographic authentication of received NTP packets. This is implemented in XNTPD by using the DES or MD5 algorithms to compute a digital signature, or message digest. The specification allows any one of possibly four billion keys, numbered with 32-bit key identifiers, to be used to authenticate an association. The servers involved in an association must agree on the key and key identifier used to authenticate their messages.

Keys and related information are specified in a key file that should be exchanged and stored by using secure procedures beyond the scope of the protocol. There are three classes of keys involved in the current implementation. One class is used for ordinary NTP associations, another is used for the NTPQ utility program, and the third is used for the XNTPDC utility program.

Table 6-11 *XNTPD Authentication Command Options*

Parameter	Description
keys *keyfile*	Specifies the filename containing the encryption keys and key identifiers used by XNTPD, NTPQ and XNTPDC when operating in authenticated mode. For ntp.keys file format see ntp.keys.
trustedkey *key* [...]	Specifies the encryption key identifiers that are trusted for the purposes of authenticating peers suitable for synchronization. The authentication procedures require that both the local and remote servers share the same key and key identifier for this purpose, although different keys can be used with different servers. The key arguments are 32-bit unsigned integers. NTP key 0 is fixed and globally known. If meaningful authentication is to be performed, the 0 key should not be trusted.
requestkey *key*	Specifies the key identifier to use with the XNTPDC program, which uses a proprietary protocol specific to this implementation of XNTPD. This program is useful to diagnose and repair problems that affect XNTPD operation. The key argument to this command is a 32-bit unsigned integer. If no requestkey command is included in the configuration file, or if the keys don't match, such requests are ignored.
controlkey *key*	Specifies the key identifier to use with the NTPQ program, which uses the standard protocol defined in RFC-1305. This program is useful to diagnose and repair problems that affect the XNTPD operation. The key argument to this command is a 32-bit unsigned integer. If no `requestkey` command is included in the configuration file, or if the keys don't match, such requests are ignored.

For DES, the keys are 56 bits long with, depending on type, a parity check on each byte. For MD5, the keys are 64 bits (8 bytes). XNTPD reads its keys from a file specified by using the -k command line option or the keys statement in the configuration file. Although the key number 0 is fixed by the NTP standard (as 56 zero bits) and cannot be changed, one or more of the keys numbered 1 through 15 can be arbitrarily set in the keys file.

The key file uses the same comment conventions as the configuration file. Key entries use a fixed format of the form

```
keyno type key
```

where `keyno` is a positive integer, type is a single character that defines the key format, and key is the key itself.

The key can be given in one of three different formats, controlled by the type character. The three key types, and corresponding formats, are listed in the following table:

Table 6-12 *XNTPD Key File Parameters*

Parameter	Description
S	The key is a 64-bit hexadecimal number in the format specified in the DES specification; that is, the high order seven bits of each octet are used to form the 56-bit key while the low order bit of each octet is given a value so that odd parity is maintained for the octet. Leading zeroes must be specified (that is, the key must be exactly 16 hex digits long) and odd parity must be maintained. A zero key, in standard format, would be given as 0101010101010101.
N	The key is a 64-bit hexadecimal number in the format specified in the NTP standard. This is the same as the DES format, except the bits in each octet have been rotated one bit right so that the parity bit is now the high order bit of the octet. Leading zeroes must be specified and odd parity must be maintained. A zero key in NTP format would be specified as 8080808080808080.
A	The key is a 1-to-8 character ASCII string. A key is formed from this by using the low order 7 bits of each ASCII character in the string, with zeroes added on the right when necessary to form a full width 56-bit key, in the same way that encryption keys are formed from UNIX passwords.
M	The key is a 1-to-8 character ASCII string, using the MD5 authentication scheme. both the keys and the authentication schemes (DES or MD5) must be identical between a set of peers sharing the same key number.

The keys used by the NTPQ and XNTPDC programs are checked against passwords requested by the programs and entered by hand, so it is generally appropriate to specify these keys in ASCII format.

6.5.5 Monitoring Options

XNTPD includes a comprehensive monitoring facility suitable for continuous, long-term recording of server and client timekeeping performance. See the statistics commands below for a listing and example of each type of statistics currently supported. Statistics files are managed by using file generation sets and scripts in the `./scripts` directory of this distribution. Using these facilities and UNIX cron jobs, the data can be automatically summarized and archived for retrospective analysis.

Table 6-13 *XNTPD Monitoring Command Parameters*

Parameter	Description
statistics *name* [...]	Enables writing of statistics records. Three kinds of *name* statistics are supported. They are loopstats, peerstats, clockstats.

Parameter	Description
loopstats	Enables recording of loop filter statistics information. Each update of the local clock outputs a line of the following form to the file generation set named loopstats:
	`48773 10847.650 0.0001307 17.3478 2`
	The first two fields show the date (Modified Julian Day) and time (seconds and fraction past UTC midnight). The next three fields show time offset in seconds, frequency offset in parts-per-million, and time constant of the clock-discipline algorithm at each update of the clock.
peerstats	Enables recording of peer statistics information. This includes statistics records of all peers of a NTP server and of special signals, where present and configured. Each valid update appends a line of the following form to the current element of a file generation set named peerstats:
	`48773 10847.650 127.127.4.1 9714 -0.001605 0.00000 0.00142`
	The first two fields show the date (Modified Julian Day) and time (seconds and fraction past UTC midnight). The next two fields show the peer address in dotted-quad notation and the status. The status field is encoded in hex format. The final three fields show the offset, delay and dispersion, all in seconds.
clockstats	Enables recording of clock driver statistics information. Each update received from a clock driver outputs a line of the following form to the file generation set named clockstats:
	`49213 525.624 127.127.4.1 93 226 00:08:29.606 D`
	The first two fields show the date (Modified Julian Day) and time (seconds and fraction past UTC midnight). The next field shows the clock address in dotted-quad notation, The final field shows the last timecode received from the clock in decoded ASCII format, where meaningful. In some clock drivers, additional information can also be gathered and displayed. See information specific to each clock for further details.
statsdir *directory_path*	Indicates the full path of a directory where statistics files should be created. This keyword allows the (otherwise constant) filegen filename prefix to be modified for file generation sets, which is useful for handling statistics logs.

Parameter	Description
filegen *name* [file *filename*] [type *typename*] [flag *flagval*] [link | nolink] [enable | disable]	Configures setting of the generation file set *name*. Generation file sets provide a means for handling files that are continuously growing during the lifetime of a server. Server statistics are a typical example for such files. Generation file sets provide access to a set of files used to store the actual data. At any time, one element of the set is being written to. The type given specifies when and how data is directed to a new element of the set. This way, information stored in elements of a file set that are currently unused is available for administrative operations without the risk of disturbing the operation of XNTPD. (In addition, they can be removed to free space for new data produced.)

Filenames of set members are built from three elements:

prefix: This is a constant filename path. It is not subject to modifications via the filegen option. It is defined by the server, usually specified as a compile-time constant. It can be configurable for individual file generation sets via other commands. For example, the prefix used with `loopstats` and `peerstats` generation can be configured using the `statsdir` option explained above.

filename: This string is directly concatenated to the prefix mentioned above (no intervening / (slash)). This can be modified by using the `file` argument to the `filegen` statement. No .. elements are allowed in this component to prevent filenames referring to parts outside the filesystem hierarchy denoted by `prefix`.

suffix: This part reflects individual elements of a file set. It is generated according to the type of a file set.

A file generation set is characterized by its type. The following types are supported:

- ◆ **none:** The file set is actually a single plain file.

- ◆ **pid:** One element of file set is used per incarnation of an XNTPD server. This type does not perform any changes to file set members during runtime; however, it provides an easy way of separating files belonging to different XNTPD server incarnations. The set member filename is built by appending a . (dot) to concatenated prefix and filename strings, and appending the decimal representation of the process ID of the XNTPD server process.

- ◆ **day:** One file generation set element is created per day. A day is defined as the period between 00:00 and 24:00 UTC. The file set member suffix consists of a . (dot) and a day specification in the form YYYYMMDD. YYYY is a 4-digit year number (e.g., 1992). MM is a two digit month number. DD is a two digit day number. Thus, all information written at 10 December 1992 would end up in a file named prefix filename.19921210.

- ◆ **week:** Any file set member contains data related to a certain week of a year. The term week is defined by computing day-of-year modulo 7. Elements of such a file generation set are distinguished by appending the following suffix to the file set filename base: A dot, a 4-digit year number, the letter W, and a 2-digit week number. For example, information from January, 10th 1992 would end up in a file with suffix .1992W1.

- ◆ **month:** One generation file set element is generated per month. The filename suffix consists of a dot, a 4-digit year number, and a 2-digit month. |

Parameter	Description
filegen *name* [file *filename*] [type *typename*] [flag *flagval*] [link | nolink] [enable | disable] (**cont'd.**)	◆ **year:** One generation file element is generated per year. The filename suffix consists of a dot and a 4-digit year number. ◆ **age:** This type of file generation sets changes to a new element of the file set every 24 hours of server operation. The filename suffix consists of a dot, the letter a, and an 8-digit number. This number is taken to be the number of seconds the server is running at the start of the corresponding 24-hour period. Information is only written to a file generation by specifying enable; output is prevented by specifying disable.

It is convenient to be able to access the current element of a file generation set by a fixed name. This feature is enabled by specifying link and disabled by using nolink. If link is specified, a hard link from the current file set element to a file without a suffix is created. When there is already a file with this name and the number of links of this file is one, it is renamed appending a dot, the letter C, and the pid of the XNTPD server process. When the number of links is greater than one, the file is unlinked. This allows the current file to be accessed by a constant name.

6.5.6 Access Control Options

XNTPD implements a general purpose address-and-mask-based restriction list. The list is sorted by address and by mask, and the list is searched in this order for matches, with the last match found defining the restriction flags associated with the incoming packets. The source address of incoming packets is used for the match, with the 32-bit address being added with the mask associated with the restriction entry and then compared with the entry's address (which has also been added with the mask) to look for a match.

The restriction facility was implemented in conformance with the access policies for the original NSFnet backbone time servers. Although this facility might be otherwise useful for keeping unwanted or broken remote time servers from affecting your own, it should not be considered an alternative to the standard NTP authentication facility. Source address based restrictions are easily circumvented by a determined cracker.

Table 6-14 *XNTPD Access Control Parameters*

Parameter	Description
ntpport	This is actually a match algorithm modifier, rather than a restriction flag. Its presence causes the restriction entry to be matched only if the source port in the packet is the standard NTP UDP port (123). Both ntpport and non-ntpport can be specified. The ntpport is considered more specific and is sorted later in the list. Default restriction list entries, with the flags ignore ntpport for each of the local host's interface addresses, are inserted into the table at startup to prevent the server from attempting to synchronize to its own time. A default entry is also always present, although if it is otherwise unconfigured; no flags are associated with the default entry (i.e., everything besides your own NTP server is unrestricted).
clientlimit *limit*	Set the *client_limit* variable, which limits the number of simultaneous access-controlled clients. The default value for this variable is 3.

Parameter	Description
clientperiod *period*	Set the *client_limit_period* variable, which specifies the number of seconds after which a client is considered inactive and thus is no longer counted for client limit restriction. The default value for this variable is 3600 seconds.

6.5.7 Miscellaneous Options

Table 6-15 *Miscellaneous XNTPD Parameters*

Parameter	Description
broadcastdelay *seconds*	
trap *host_address* [port *port_number*] [interface *interface_address*]	This command configures a trap receiver at the given host address and port number for sending messages with the specified local interface address. If the port number is unspecified, a value of 18447 is used. If the interface address is not specified, the message is sent with a source address of the local interface the message is sent through.
	NOTE: On a multihomed host, the interface used can vary from time to time with routing changes.
	The trap receiver will generally log event messages and other information from the server in a log file. While such monitor programs may also request their own trap dynamically, configuring a trap receiver will ensure that no messages are lost when the server is started.
setvar *variable* [default]	This command adds an additional system variable. These variables can be used to distribute additional information such as the access policy. If the variable of the form *name = value* is followed by the default keyword, the variable is listed as part of the default system variables (NTPQ rv command). These additional variables serve informational purposes only. They are not related to the protocol other than being listed. The known protocol variables always override any variables defined via the setvar mechanism.
	There are three special variables that contain the names of all variables of the same group. The *sys_var_list* holds the names of all system variables. The *peer_var_list* holds the names of all peer variables and the *clock_var_list* holds the names of the reference clock variables.
logfile *logfile*	This command specifies the location of an alternate log file to be used instead of the default system syslog facility.

6.5.8 Variables

Most variables used by the NTP protocol can be examined with the XNTPDC (mode 7 messages) and the NTPQ (mode 6 messages). Currently, very few variables can be modified via mode 6 messages. These variables are either created with the setvar directive or the leap warning bits. The leap warning bits can be set in the leapwarning variable up to one month ahead. Both the leapwarning and leapindication variables have a slightly different encoding than the usual leap bits interpretation:

- ◆ **00:** The daemon passes the leap bits of its synchronization source (usual mode of operation).

- **01/10:** A leap second is added/deleted (operator forced leap second).
- **11:** Leap information from the synchronization source is ignored (so LEAP_NOWARNING is passed on).

Monitoring and Security

7

This section describes NTP monitoring and security.

7.1 Monitoring Time Synchronization

The quality of time synchronization can be monitored. It is based on the accuracy of the time provided by the time provider to the time consumer.

The time quality variables like offset, jitter, and precision can be measured and logged for online or offline analysis in the text mode for any NTPv3-compliant operating system.

7.1.1 Using NTPQ to Monitor Time Quality

The NTPQ utility can be used to monitor time quality variables consists of the following commands:

Table 7-1 *NTPQ commands*

Command	Description
peers	Obtains a list of in-spec peers of the server, along with a summary of each peer's state. Summary information includes the remote peer's address, reference ID (0.0.0.0 if the refID is unknown), stratum, type (local, unicast, multicast, or broadcast); when the last packet was received; the polling interval (in seconds); the leachability register (in octal); and the current estimated delay, offset, and dispersion of the peer (all in seconds).
associations	Obtains and displays a list of association identifiers and peer statuses for in-spec peers of the server being queried. This command can also be used to check if the server is synchronized or not (If the Status column contains sys.peer, it means that server is synchronized).
rv *associate*	Obtains various time parameters from the server and displays them. The most important parameter is filter offset, which gives the last eight time offsets of the host server with the reference server.
rvi *index*	Obtains various time parameters from the server and displays them. The sequencing index can be used instead of *assocID* (as in the `rv` command). This makes it easy to monitor and remember the association ID every time.

For more information about the commands, see Section 6.2, "NTPQ," on page 38.

The following figure displays the output of the NTPQ peers, association, and `rv` `assocID` commands.

Figure 7-1 *Output of NTPQ Peers, Association, and RV AssocID Commands*

The following figure displays the output of the NTPQ `rvi index` command.

Figure 7-2 *Output of the NTPQ RVI Index Command*

7.1.2 Using the Health Monitor to Monitor NTP

You can monitor NTP remotely by using Novell® Remote Manager. You can monitor all the servers that are authenticated to the Novell eDirectory™ 8.7.3 or later tree. Select the server you want to monitor. The Peer, Associations, rv, and rvi details are displayed. These details are similar to output that the NTPQ peers, NTPQ associations and NTPQ read variables commands (rv and rvi) give. For more information, see .

Figure 7-3   *Monitoring NTP by Using Novell® Remote Manager..*

7.2 Security

XNTPD supports the optional authentication procedure specified in NTP versions 2 and 3.

When an association runs in the authenticated mode, each packet transmitted appends a 32-bit key ID and a 64/128-bit cryptographic checksum of the packet contents. This is computed using either the Data Encryption Standard (DES) or Message Digest (MD5) algorithms. These algorithms provide sufficient protection from message modification attacks.

NOTE: Distribution of DES algorithm implementation is restricted to U.S. and Canada, but MD5 is currently free from such restrictions.

In both the algorithms, the receiving peer recomputes the checksum and compares it with the one included in the packet. For this to work, the peers must share at least one encryption key and must associate the shared key with the same key ID.

This requires some minor modifications to the basic packet processing procedures, as required by the specification. These modifications are enabled by the "enable authenticate" configuration declaration.

In particular, the following servers are marked untrustworthy and unsuitable for synchronization in the authenticated mode:

- Peers that send unauthenticated packets
- Peers that send authenticated packets that the local server is unable to decrypt
- Peers that send authenticated packets encrypted by using a key that NTP does not trust

---

NOTE: Although the server might know many keys (identified by many key IDs), it is possible to declare only a subset of these as trusted. This allows the server to share keys with a client that trusts the server and requires authenticated time, even though the server does not trust the client.

Also, some additional configuration language is required to specify the key ID to be used to authenticate each configured peer association. For example, for a server running in authenticated mode, the configuration file might look similar to the following:

```
# peer configuration for 128.100.100.7

# (expected to operate at stratum 2)

# fully authenticated this time

peer 128.100.49.105 key 22 # suzuki.ccie.utoronto.ca

peer 128.8.10.1 key 4     # umd1.umd.edu

peer 192.35.82.50 key 6  # lilben.tn.cornell.edu

enable auth

keys sys:\etc\ntp.keys    # path for key file

trustedkey 22 4 6     # define trusted keys

requestkey 15              # key (6) for accessing server variables

controlkey 15              # key (7) for accessing server variables

authdelay 0.000094        # authentication delay (Sun4c/50 IPX)
```

- The `enable auth` line enables authentication processing.
- The `keys sys:\etc\ntp.keys` line specifies the path to the keys file (see below and the XNTPD document page for details of the file format).
- The `trustedkey` line identifies those keys that are known to be uncompromising; the remainder presumably represent the expired or possibly compromised keys. Both sets of keys must be declared by the key identifier in the `ntp.keys` file described below. This provides a way to retire old keys while minimizing the frequency of delicate key-distribution procedures.
- The `requestkey 15` line establishes the key to be used for mode 6 control messages as specified in RFC-1305 and used by the NTPQ utility program.
- The `controlkey 15` establishes the key to be used for mode 7 private control messages used by the XNTPDC utility program. These keys are used to prevent unauthorized modification of daemon variables.

As a general rule, keys should be chosen randomly, except possibly the request and control keys, which must be typed by the user as a password.

ntp.keys

The `ntp.keys` file contains the list of keys and associated key IDs that the server knows about.

WARNING: This file should be left unreadable by anyone except admin.

The contents of the `ntp.keys` file might look similar to the following:

```
# ntp keys file (ntp.keys)

1    N    29233E0461ECD6AE    # des key in NTP format

2    M    RIrop8KPPvQvYotM    # md5 key as an ASCII random string

14   M    sundial             # md5 key as an ASCII string

15   A    sundial             # des key as an ASCII string

# the following 3 keys are identical

10   A    SeCReT

10   N    d3e54352e5548080

10   S    a7cb86a4cba80101
```

Each line in the key file has three attributes. For example:

```
1    N    29233E0461ECD6AE
```

In the above example:

  ◆ 1 is the key ID
  ◆ N is the key format
  ◆ 29233E0461ECD6AE is the key itself

The following table explains the four key formats.

Table 7-2 *XNTPD Security Key Formats*

Key Format	Description
A	Indicates a DES key written as a 1-to-8 character string in 7-bit ASCII representation, with each character standing for a key octet (like a UNIX password).
M	Indicates an MD5 key written as a 1-to-31 character ASCII string in the A format.
N	Indicates a DES key again written as a hex number, but in the NTP standard format with the high order bit of each octet being the (odd) parity bit.
S	Indicates a DES key written as a hex number in the DES standard format, with the low order bit (LSB) of each octet being the (odd) parity bit.

NOTE: Because of the simple token routine, the characters #, \t, \n, \0 and a space cannot be used in either a DES or MD5 ASCII key. Key 0 (zero) is used for special purposes and should not appear in this file.

7.2.1 Sample Scenario

This sample scenario demonstrates a setup where XNTPD on ServerB (time consumer) needs to take time from the XNTPD on ServerA (time provider) with authentication.
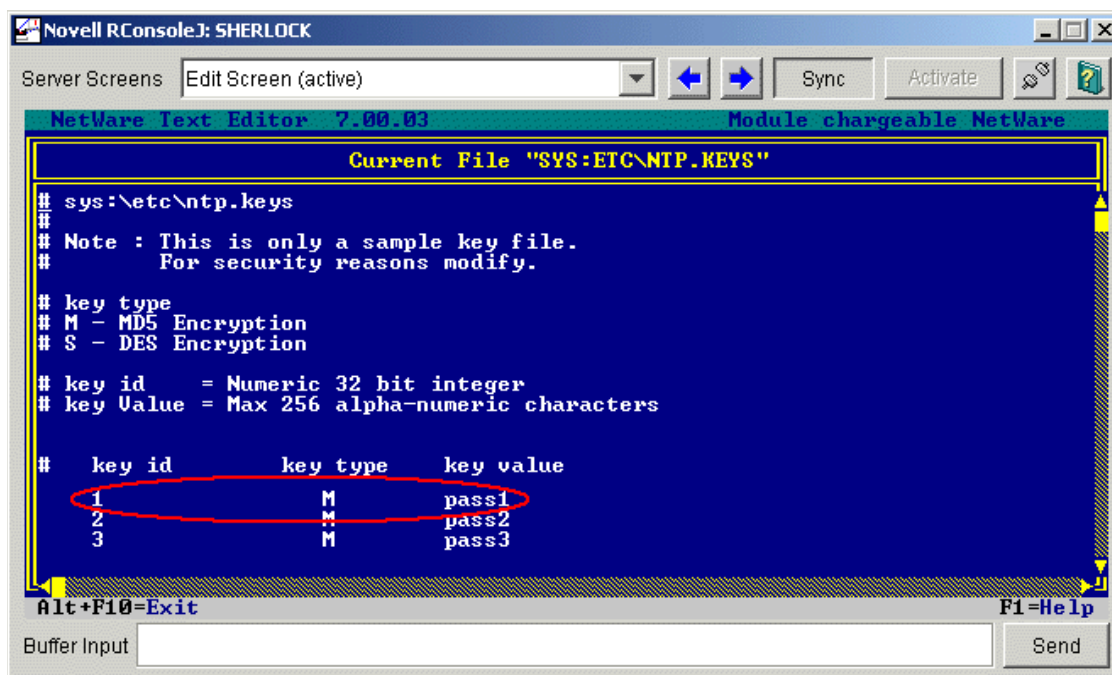
On ServerA:

- In the `ntp.conf` file located in `sys:\etc`, make the following changes:
  - Mention `sys:\etc` as the path of the key file as follows:

    `keys sys:\etc\ntp.keys`
  - Mark 1 as the trusted key ID. See the following figure for more details.

Figure 7-4 *Ntp.conf File of ServerA*
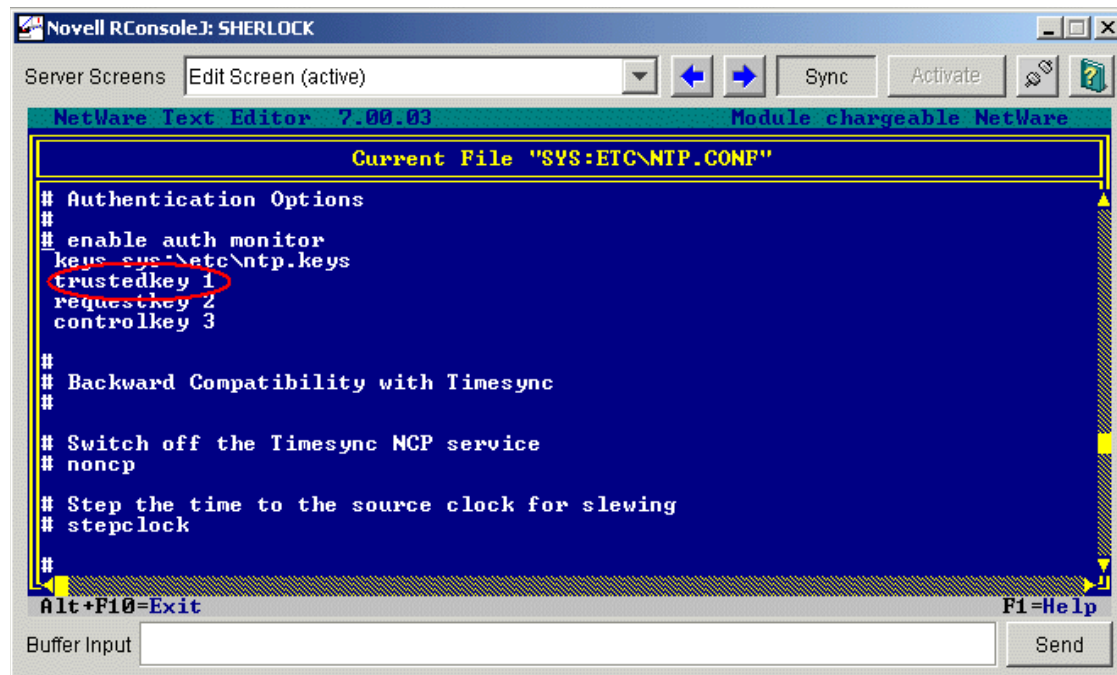
- In the `ntp.keys` file located in `sys:\etc`, give a key value for key ID 1, pass1.
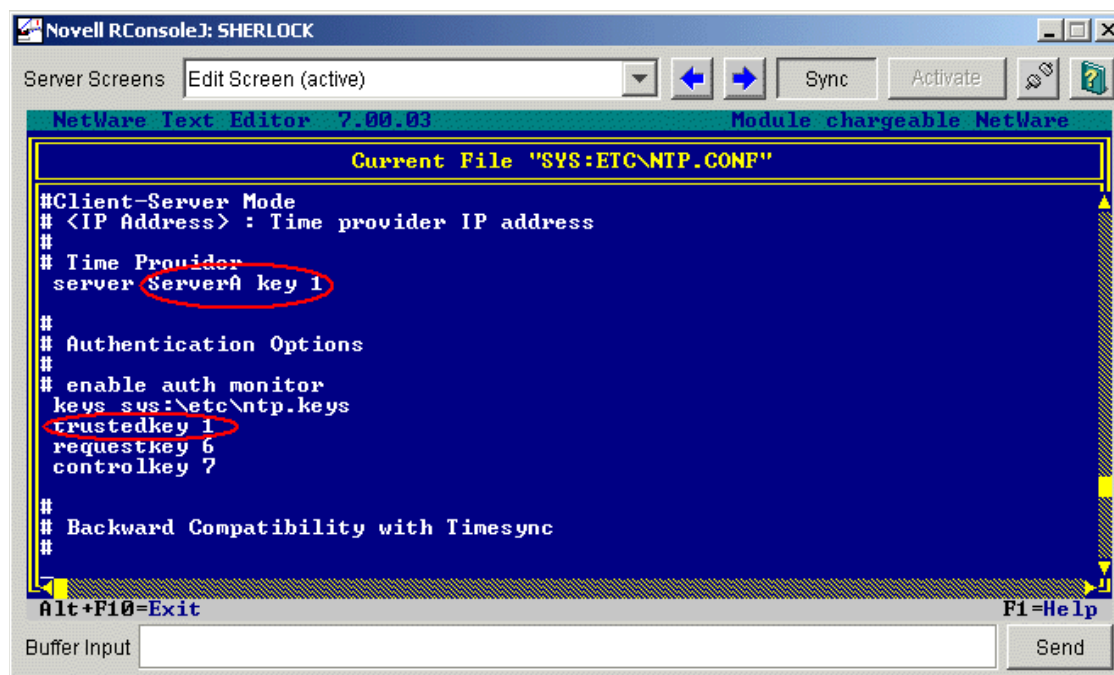
Figure 7-5 *Ntp.keys File of ServerA*

On ServerB:

- In the `ntp.conf` file located in `sys:\etc`, make the following changes:
    - Mention `sys:\etc` as the path of the key file as follows:

        `keys sys:\etc\ntp.keys`

    - Mark 1 as the trusted key ID. See the following figure for more details.

Figure 7-6   *Ntp.conf File of ServerB*

- In the `ntp.keys` file located in `sys:\etc`, give the same key value (that was given in ServerA for key ID 1) as shown in the following figure.

Figure 7-7   *Ntp.keys File of ServerB*
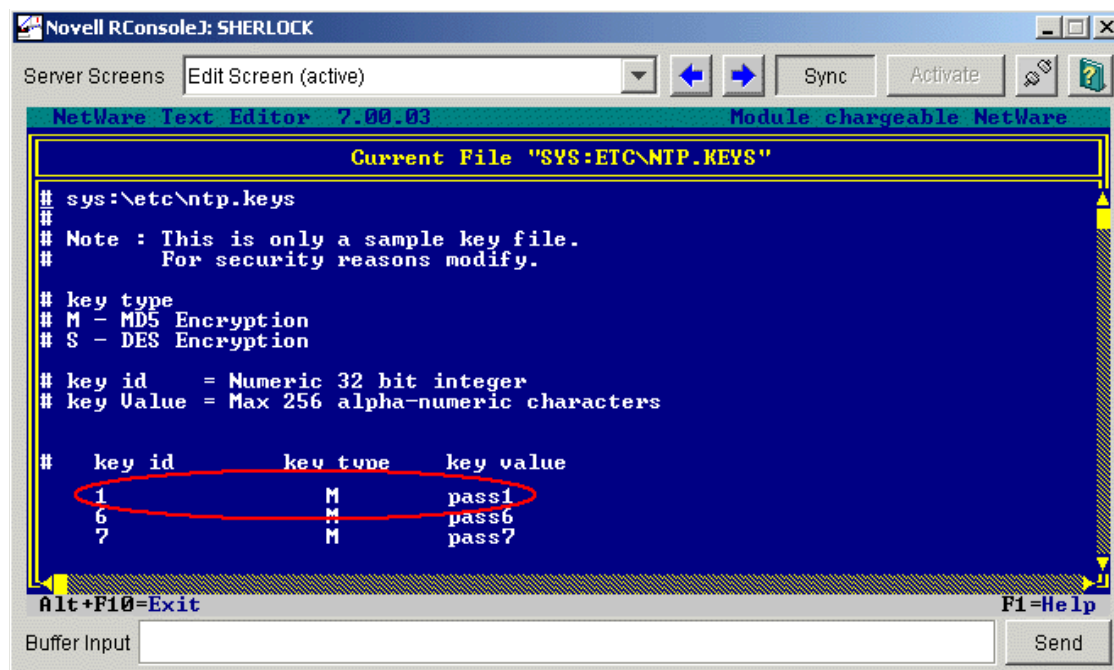
After entering the details in ServerA's and ServerB's `ntp.conf` and `ntp.keys` files, restart XNTPD on both the servers.

Migrating Timesync/NTP from NetWare to NTP on OES 2 Linux

8

The OES 2 SP1 Migration Tool has a plug-in architecture and is made up of Linux command line utilities with a GUI wrapper. You can migrate CIFS from a NetWare server to an OES 2 SP1 Linux server either using the GUI Migration Tool or from the command line.

To get started with migration, see *OES 2 SP2: Migration Tool Administration Guide*

For more information on migrating NTPv3, see "Migrating Timesync/NTP from NetWare to NTP on OES 2 Linux".

Troubleshooting NTP

<div style="text-align: right; font-size: 3em;">9</div>

The following sections give suggestions and resources for solving issues with NTP:

9.1 XNTPD

This section provides solutions to problems you might encounter when using XNTPD.

XNTPD -T slp does not update ntp.conf

Problem: You are looking for a server that is advertising its Timesync SINGLE time source service that is in a different tree.

Action: To verify this:

1. Enter `display slp services timesync.novell` at the server console to display a list of servers advertising the Timesync SINGLE service.

2. If you know the tree name of your server, then enter `display slp services timesync.novell://treename==`*mytreename*.

XNTPD broadcast functionality is not working

Problem: You are using an incorrect subnet broadcast ID.

Action: Ensure that you have specified the correct subnet broadcast address.

Action: Use `showipconf` in NTPQ.

Unable to configure XNTPD remotely using XNTPDC

Problem: You do not have the proper keys to authenticate to the server.

Action: Ensure that you have the request key ID and the key of the server you are trying to configure. You can obtain the request key ID from the ntp.conf file and the request key from the ntp.keys file. Both files are located in the sys:\etc directory.

XNTPD cannot obtain time in the broadcast/multicast mode

Problem: You have not authenticated to broadcast/multicast server.

Action: Start `XNTPD` with the -*A* option. This disables authentication.

Action: Obtain the time provider's key, copy it to `ntp.keys` file, and mark it as a trusted key in `ntp.conf`. Ensure that this key is present in the broadcast/multicast server.

XNTPD takes a long time to synchronize

Problem: The polling interval is too big, so the polls are too far apart. XNTPD needs 5 successful polls (offset lesser then 150 ms) to transit into synchronized mode. By default, XNTPD has the minpoll value set to 4. This means that every poll is 2 power 4 seconds away. Hence, XNTPD syncs in, at best, 5 * (2 power 4) seconds.

Action: Minimize the polling interval by using minpoll. Set minpoll equal to 4 in the `ntp.conf.` file while giving the time provider details. For example: `server` *IP address* `minpoll 4`

Action: Use XNTPD with the -S option or stepclock in the `ntp.conf.` file to synchronize within 10 seconds. For more information, see "-S" on page 46 or "stepclock" on page 51.

Unable to enable the debug message with XNTPD

Problem: Unable to get the debug message with XNTPD.

Action: To enable the debug message with XNTPD, enter the following at the command prompt:

```
Load XNTPD -D 4
```

Time does not synchronize if XNTPD is loaded without the -A parameter in the multicast mode

Problem: In the multicast mode, time is not synchronized if XNTPD is loaded without the `-A` parameter.

Action: If you load `XNTPD` with the *-A* parameter, it means that you are loading XNTPD with authentication disabled.

By default, in the broadcast/multicast mode, XNTPD starts with authentication enabled. This is because XNTPD can obtain time only from a server it trusts and the trust is achieved only through authentication.

If there are no masquerading servers in the subnet/network, you can start `XNTPD` with authentication disabled (*-A* option).

Action: Obtain the time provider's key, copy it to the `ntp.keys` file, and mark it as a trusted key in the `ntp.conf` file. Ensure that this key is present in the broadcast/multicast server.

Time does not synchronize in the broadcast/multicast mode it takes time from the local clock

Problem: A local clock is not a very reliable time source and it cannot be broadcast or multicast.

Action: Use the `prefer` keyword when using a local clock as follows:
`server 127.127.1.0 prefer`

XNTPD hangs while loading

Action: Ensure that XNTPD, NTPQ, NTPDate, and XNTPDC have unloaded successfully.

XNTPD goes out of synch frequently

      Action: Run XNTPD with the -S option

XNTPD exits after 5 to 10 minutes

      Problem: The time offset between the client and server is too big, so XNTPD does not adjust the clock and it exits.

      Action: Start XNTPD with the -S option.

9.2 NTPDate

This section provides solutions to problems you might encounter when using NTPDate.

NTPDate loads normally, but does not set the clock

      Action: All the error messages appear on the Logger screen; check the error details.

Unable to load NTPDate when Timesync / XNTPD is running

      Problem: NTPDate was unable to bind to port 123.

      Action: Load NTPDate with the -u option.

9.3 General

This section provides solutions to generic problems you might encounter.

Unable to configure NTPv3 as time client

      Action: Make sure the time provider is broadcasting/multicasting its service.

      Action: Make sure time sources are in synchronization.

      Action: If you have localhost as loopback, ensure that you specify a higher stratum for the loopback so that it has lower precedence than the other time source.

Unable to open the ntp.log file using Edit

      Action: You might not be able to open the `ntp.log` file if it was created for the first time. Subsequent loads of XNTPD would not cause this issue.

Frequently Asked Questions

10

This section lists frequently asked questions and their answers for NTPv3.

10.1 How Do I Configure NTP for Fault Tolerance?

Configure two servers as follows.

For Server1:

- Configure XNTPD to obtain time from the external NTP time source and mark it as Prefer.
- Add a local clock as the second server with a stratum value greater than external NTP source.

```
server IP_address_of_external_time_source-1 prefer

server 127.127.1.0

fudge 127.127.1.0 stratum 4
```

For Server2:

- Configure XNTPD to obtain time from the external NTP time source and mark it as Prefer.
- Add Server1 as peer to Server2.

```
server IP_address_of_external_time_source-2 prefer

peer IP_address_of_Server1
```

Figure 10-1   *Fault Tolerance Configuration*
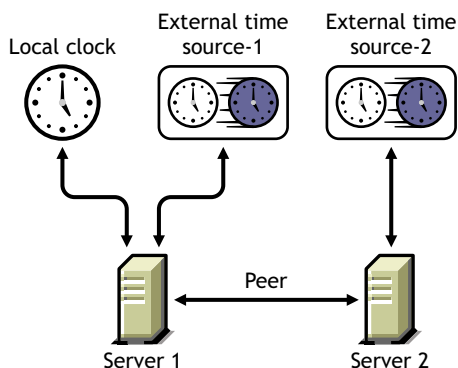
Server1 and Server2 provide time to all other servers in the network.

10.2 How Do I Find the Subnet Broadcast Address?

To find the subnet broadcast address, do one of the following:

- Use `showipconf` in NTPQ.
- Start XNTPD with debug level 4 and the log option enabled. XNTPD displays the subnet broadcast address for each time source.

10.3 How Do I Find which Server Is Selected for Synchronization?

To view the server selected for synchronization, do one of the following:

- Run NTPQ with the server name.
- Enter the command `pe`.

  The server selected for synchronization is shown with an asterisk (*).

10.4 How Do I Find the Offset Value between the Client and the Server?

Do one of the following:

- Enter `ntpdate -d` *server_name* at command prompt. This lists the offset.
- Run NTPQ with the client and enter `pe`. This lists the offset of the client with each time source.

10.5 Can a Single Timesync Server Take Time from a NetWare 6.5 Server Running NTPv3?

Yes, a Single Timesync server can take time from a NetWare 6.5 server running NTPv3 in both NTP and NCP modes.

Known Issues

<div style="text-align: right; font-size: 3em;">A</div>

This section contains the known issues of NTP.

- Only the NetWare 6.5 servers in the same Novell® eDirectory™ 8.7.3 or later tree is configured.

- By default, Timesync is loaded with the NetWare® 6.5 installation. To make XNTPD load by default, edit the `sys:\system\timeserv.ncf` file

Documentation Updates

B

B.1 November 9, 2009

This guide has been modified for publication on the NetWare 6.5 SP8 Documentation Web site.

B.2 December 2008

- Changed references of eDirectory™ 8.7.3 to eDirectory 8.7.3 or later.
- Changed references of iManager to iManager 2.7.1.
- Revised the Migration section and moved it to the *OES 2 SP2: Migration Tool Administration Guide*.
- Added the Virtualization Chapter.
- Updated the front file to the latest template structure.
- Edited the content for minor changes such as SubToc.
- Changed the graphic file format to png.

B.3 May 9, 2005

- Changed references of iManager 2.0 to iManager 2.5.
- Changed references of eDirectory™ to eDirectory 8.7.3.
- Added a section with Documentation Updates information.