Novell Network Time Protocol

1.0

www.novell.com

ADMINISTRATION GUIDE

Novell®

Novell, Inc.
1800 South Novell Place
Provo, UT 84606
U.S.A.

www.novell.com

Network Time Protocol Administration Guide
April 2003

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see www.novell.com/documentation.

Contents

About this Guide

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

Documentation Updates

For the most recent version of this *Network Time Protocol Administration Guide*, see the Novell documentation http://www.novell.com/documentation/beta/nw65/ntp/data/aizwub2.html (http://www.novell.com/documentation/beta/nw65/ntp/data/aizwub2.html) Web site.

Documentation Conventions

All occurrences of NTP in this chapter refer to NTP version 3 or 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.

1 Network Time Protocol

The Network Time Protocol (NTP) is used to synchronize servers, which are NTPv3 compliant. Operating systems like UNIX*, Windows* NT* and NetWare® are NTPv3 compliant.

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

- www.ntp.org (http://www.ntp.org)
- www.eecis.udel.edu/~mills (http://www.eecis.udel.edu/~mills)

NTP Terminology

This documentation uses the following terms:

- **NTP time provider:**

  A server that understands 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,

  - 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 "Broadcast or Multicast Mode" on page 14.

- **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 "Client-Server Mode" on page 13, "Peer-to-Peer Mode" on page 14, and "Broadcast or Multicast Mode" on page 14.

  The time consumer can work with operating systems that are NTP and NCP compliant, like:

  - 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 due to fluctuations in the voltage supply, mechanical vibrations, or other sources.

- **Root delay:**

  The total roundtrip 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 on 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 and is 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.

NTP Implementation on NetWare

Novell® eDirectory™ relies heavily on consistent and reliable time stamps for its objects. Without such time stamps, synchronization of directories is not possible.

NTP addresses fault tolerance in the form of 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 reduces, minimizing traffic across routers and WANs.

Features of NTP on NetWare

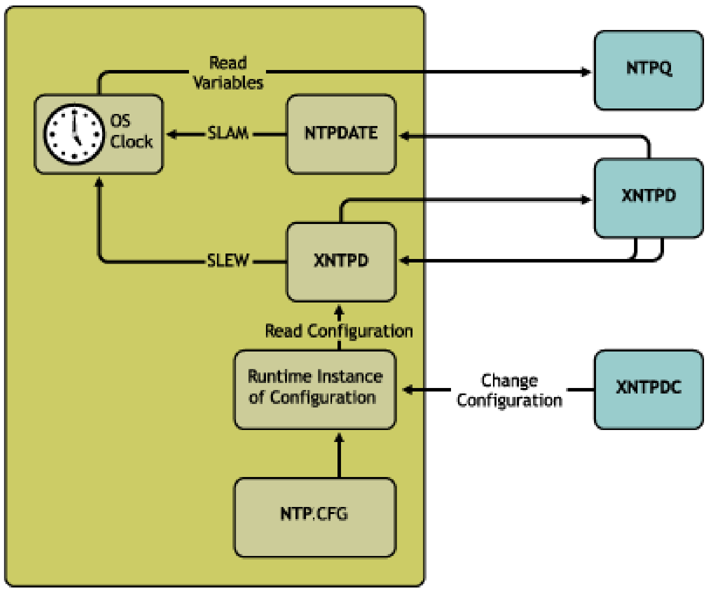
- All features as per RFC 1305 on NTPv3.

- Support for browser-based configuration (NORM). This helps in centralizing the support and configuration of NTP time synchronization service on the network for the complete eDirectory tree.
- Support of backward compatibility for Timesync.

NTP Architecture

This section briefly outlines the NTP architecture.

Figure 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** is the remote configuration utility. It is used to query the XNTPD daemon about its current state and to request changes in that state.
- **XNTPD** is an operating system daemon which sets and maintains the system time-of-day in synchronism with Internet standard time servers.

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 adjusts the local clock to the time of the other server gradually. XNTPD sets and maintains the time on the local clock.

Ntp.cfg 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.cfg file.

2 Modes of Time Synchronization

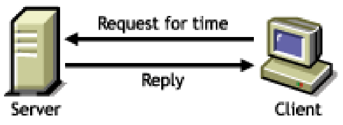
You can synchronize time using the following three methods:

- "Client-Server Mode" on page 13
- "Peer-to-Peer Mode" on page 14
- "Broadcast or Multicast Mode" on page 14

Client-Server Mode

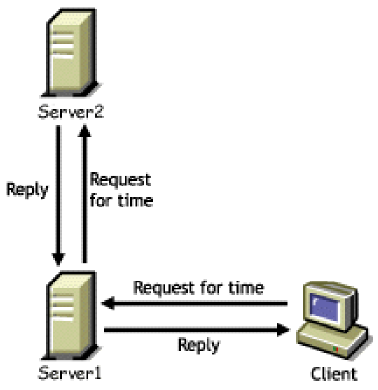
With this mode, the time consumer requests the time provider for the time and the time provider sends back the time taking into account the time delays and other contingencies.

Figure 2 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 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.cfg file (located in sys:\etc):

```
server 127.127.1.0 minpoll 4
```

```
fudge server 127.127.1.0 stratum 10
```

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.
Stratum	Refers to the distance a host running the XNTPD time daemon is from an external source of Coordinated Universal Time (UTC). A stratum 1 server has direct access to an external source of UTC. In general, a stratum n server is n-1 network hops away from a stratum 1 server. Also, a stratum n server is at a higher stratum than a stratum n-1 server.

- 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.cfg file:

  ```
  server IP_address_of_time_provider minpoll 4
  ```

  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.cfg file:

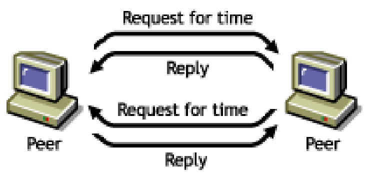
  ```
  server IP_address_of_server minpoll 4
  ```

See "Manual Configuration" on page 17 for more information.

Peer-to-Peer Mode

With 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 4 Peer-to-Peer Mode



If you want to use the servers in this mode, add the following lines to the ntp.cfg file of each server:

```
peer IP_address_of_the_other_server minpoll 4
```

See "Wide Area Configuration" on page 22 for more information.

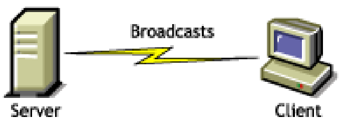
Broadcast or Multicast Mode

With 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 5 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 would listen to the advertisement, register for the service, and use it.

Broadcast

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

```
broadcastclient subnet_broadcast_address
```

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

Multicast

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

```
multicastclient
```

See “Auto Configuration” on page 20 for more details.

3 NTP Configuration

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:

- "Manual Configuration" on page 17
- "Auto Configuration" on page 20
- "Wide Area Configuration" on page 22

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.

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

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:

Planning the Setup
Configuring the Time Providers
Configuring the Time Consumers

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.

Configuring the Time Providers

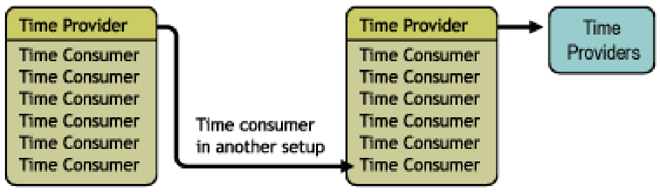
In manual configuration, a time provider can get time

- From another time provider (client-server mode)
- From its local clock (self-synchronized mode)

Client-Server Mode

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

Figure 6 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.cfg file:

   ```
   server IP_address_of_time_provider minpoll 4 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.cfg file is the configuration file for NTP. This file is located in sys:\etc. For more information, see "The Configuration File" on page 35.

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 "Client-Server Mode" on page 13.

Self-Synchronized Mode

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

Figure 7 Time Group with Self-Synchronization

To configure the time provider, do the following:

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

   ```
   server 127.127.1.0 minpoll 4 prefer
   fudge 127.127.1.0 stratum 10
   ```

2. Load XNTPD for the changes to take effect.

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

   ```
   Load XNTPD
   ```

Configuring the Time Consumers

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

   ```
   server IP_address_of_time_provider minpoll 4 prefer
   ```

2. Load XNTPD for the changes to take effect.

   To do this, enter the following at command prompt:

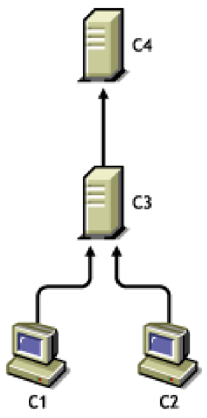
   ```
   Load XNTPD
   ```

Sample Scenario

This sample scenario demonstrates how to configure a time-synchronized setup in the manual mode.

Consider the scenario in the following figure.

Figure 8 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 configure the setup explained in the scenario using manual configuration:

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

   ```
   server IP_address_of_C3 minpoll 4 prefer
   ```

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

   ```
   server IP_address_of_C4 minpoll 4 prefer
   ```

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

   ```
   server 127.127.1.0 minpoll 4 prefer
   fudge 127.127.1.0 stratum 10
   ```

4. Load XNTPD for the changes to take effect.

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

   ```
   Load XNTPD
   ```

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:

Planning the Setup
Configuring the Time Providers
Configuring the Time Consumers

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.

Configuring the Time Provider

Broadcast

Add the following line in the time provider’s ntp.cfg 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.cfg file (located in sys:\etc) to multicast the time synchronization service on the network:

```
broadcast 224.0.1.1 key key_ID
```

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.cfg 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.cfg file.

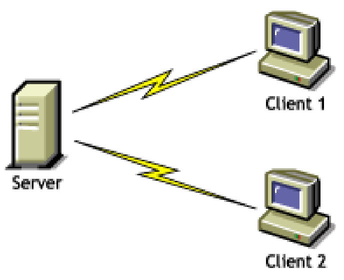
```
multicastclient
```

Sample Scenario

This sample scenario demonstrates how to configure a setup using auto configuration.

Consider the scenario in the following figure.

Figure 9 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 configure the setup in this scenario using auto configuration:

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

   ```
   broadcast subnet_broadcast_address key key_ID
   ```

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

   ```
   broadcastclient subnet_broadcast_address
   ```

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

3. Load XNTPD for the changes to take effect.

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

   ```
   Load XNTPD
   ```

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 which need fault tolerance.

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

Planning the Setup

Configuring the Time Provider Group

Configuring the Time Consumers

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.

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

To set 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.cfg file:

   ```
   peer IP_address_of_peer minpoll 4
   ```

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.cfg file:

   ```
   server IP_address_of_own_external_time_source minpoll 4
   ```

   If one external time source link goes down, both time providers would 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.cfg file:

   ```
   server 127.127.1.0

   fudge 127.127.1.0 stratum 10
   ```

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 "Peer-to-Peer Mode" on page 14 and "Client-Server Mode" on page 13.

Configuring the Time Consumers

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 "Client-Server Mode" on page 13 and "Broadcast or Multicast Mode" on page 14.

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.cfg file:

```
server IP_address_of_time_provider1_within_same_network minpoll 4
```

```
server IP_address_of_time_provider2_within_same_network minpoll 4
```

or

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

```
broadcastclient subnet_broadcast_address
```

or

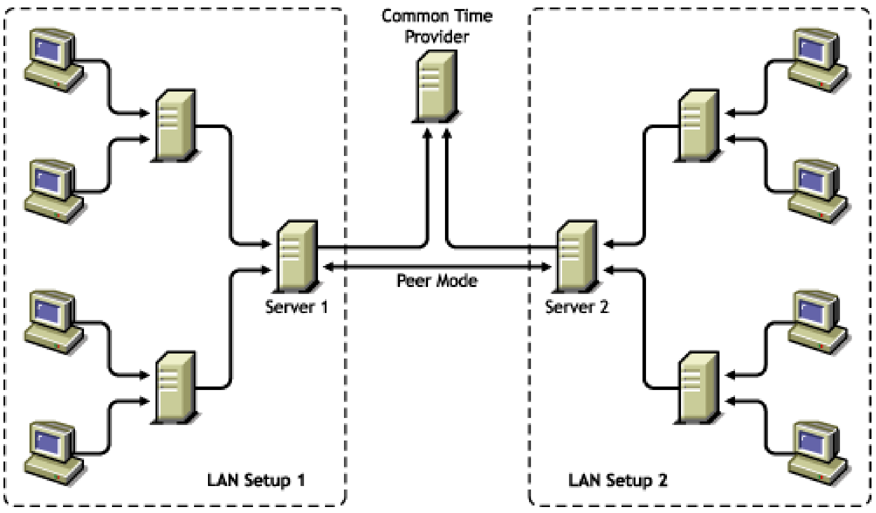
```
multicastclient
```

Sample Scenario

This sample scenario demonstrates how to configure a setup using wide area configuration.

Consider the scenario in the following figure.

Figure 10 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 "Manual Configuration" on page 17 and "Auto Configuration" on page 20 for more information.

- In both LAN setups, only Peer1 and Peer2 obtain time from sources outside the local network.
- Peer1 and Peer2 are time consumers that
  - Firstly, obtain time from a common time provider in the client-server mode.
  - Secondly, obtain time from each other in the peer mode.
  - Thirdly, fall back to their local clock if the external time sources fail.

To configure the setup in this scenario using auto configuration:

1. Configure Peer1 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 Peer1's ntp.cfg file:

   ```
   server IP_address_of_Common_Time_Provider minpoll 4 prefer
   ```

2. Configure Peer1 to obtain time from its peer, Peer2 in the peer mode.

   ```
   peer IP_address_of_Peer1 minpoll 4
   ```

3. Configure Peer1 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 will fall back to its local clock only when the other sources fail.

   ```
   server 127.127.1.0 minpoll 4
   ```

   ```
   fudge 127.127.1.0 stratum 10
   ```

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 Peer2's ntp.cfg file, to configure Peer2.

You are minimizing network traffic by configuring only Peer1 and Peer2 to obtain time from a time provider outside the local network.

Synopsis

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

Table 1 Synopsis

Configuration Type	Time Provider	Time Consumer	Best-Suited Scenario
Manual configuration for client-server mode	`server` *`IP_address_of_time_provider`* `minpoll 4 prefer`	`server` *`IP_address_of_time_provider`* `minpoll 4 prefer`	◆ Fewer than 15 servers ◆ In the same geographical location

Manual configuration for self-synchronization	`server 127.127.1.0 minpoll 4 prefer`
	`fudge 127.127.1.0 stratum 10`
		N/A	N/A

Auto configuration for broadcast	`broadcast` *`subnet_broadcast_address`* `key` *`key_ID`*	`broadcastclient` *`subnet_broadcast_address`* **NOTE:** The *subnet broadcast address* variable is optional.	◆ 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

Configuration Type	Time Provider	Time Consumer	Best-Suited Scenario

Wide area configuration for peer mode or a combination of more than one mode with fault tolerance	`server` *`IP_address_of_external_time_source`* `minpoll 4`
	
	`peer` *`IP_address_of_peer`* `minpoll 4`
	
	`server 127.127.1.0 minpoll 4`
	
	`fudge 127.127.1.0 stratum 10`
		Either manual or auto type can be used:
		
		`server` *`IP_address_of_time_provider1_within_same_network`* `minpoll 4`
		
		`server` *`IP_address_of_time_provider_2_within_same_network`* `minpoll 4`
		
		OR
		
		`broadcastclient` *`subnet_broadcast_address`*
		
		NOTE: The *subnet broadcast address* variable is optional.
			◆ Servers span across geographical locations.
			◆ Servers are synchronized in a combination of more than one mode.
			◆ Need for fault tolerance and minimizing network traffic.

Remote Configuration

NTP can be configured remotely from NetWare® Remote Manager (NORM).

NORM displays the configuration file (ntp.cfg) of all the servers in the eDirectory™ tree that you have authenticated to.

You have the following options after making changes to the ntp.cfg 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, pressing Apply for the server running Timesync, will unload Timesync and load XNTPD.
- **Restart** restarts XNTPD without saving the changes.

4 NTP Commands for Time Synchronization

This chapter explains the following NTP commands necessary to synchronize time:

NTPDate

NTPDate is used to set the local time and date using NTP servers. The NTP servers are determined by specifying them in the NTPDate command line argument. NTPDate must be run as root on the local host.

Usage:

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

Table 2 NTPDate Parameters

Parameter	Description
-a *key*	Enables the authentication function and specifies the key identifier to be used for authentication as the argument key NTPDate. 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 using the **`adjtime()`** system call, even if the measured offset is greater than +-128 ms. The default is to step the time using **`settimeofday()`** if the offset is greater than +-128 ms. **NOTE:** If the offset is greater than +-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 using the **`settimeofday()`** system call, rather than slewed (default) using the **`adjtime()`** system call. This option should be used when called from a startup file at boot time.

Parameter	Description
-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 printed.
-e *authdelay*	Specifies the processing delay to perform an authentication function as the value authdelay, in seconds and fraction (see "XNTPD" on page 34 for details). This number is usually small enough to be negligible for most purposes, though specifying a value might improve timekeeping on very slow CPUs.
-k *keyfield*	Specifies the path for the authentication key file as the string keyfile. The default is **`sys:\etc\ntp.key`**. 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. The default is 3. This allows NTPDate to be used with older NTP versions.
-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*	Diverts logging output from the standard output (default) to the *logfile*.
-t *timeout*	Specifies the maximum time waiting for a server response as the value timeout, in seconds and fraction. The value is rounded to a multiple of 0.2 seconds. The default is 1 second, a value suitable for polling across a LAN.
-u	Directs NTPDate to use an unprivileged port or outgoing packets. This is most useful when behind a firewall that blocks incoming traffic to privileged ports and you want to synchronize with hosts beyond the firewall. Note that the -d option always uses unprivileged ports.
-v	Be verbose. This option logs NTPDate's version identification string.

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 3 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 will be 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	Prints a list of the peers known to the server as well as a summary of their state.

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 may be sent to a file by appending a "<", followed by a file name, 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 following.

Table 4 Parameter Description

Parameter	Description
? [*command_keyword*] helpl [*command_keyword*]	A "?" by itself will print a list of all the command keywords known to this incarnation of NTPQ. A "?" followed by a command keyword will print 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, while the clearlist command removes all variables from the list.
authenticate yes | no	Normally NTPQ does not authenticate requests unless they are write requests. The command authenticate yes causes NTPQ to send authentication with all requests it makes. Authenticated requests causes some servers to handle requests slightly differently, and can occasionally melt the CPU in fuzzballs if you turn authentication on before doing a peer display.
cooked	Causes output from query commands to be "cooked". Variables which are recognized by the server will have their values reformatted for human consumption. Variables which NTPQ thinks 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 timestamps included in requests which require authentication. This is used to enable (unreliable) server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. Actually the server does not now require timestamps in authenticated requests, so this command may be obsolete.
host *hostname*	Set the host to which future queries will be sent. Hostname may be either a host name or a numeric address.
hostnames [yes | no]	If "yes" is specified, host names are printed in information displays. If "no" is specified, numeric addresses are printed 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.

Parameter	Description
ntpversion 1 | 2 | 3	Sets the NTP version number which 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 which demand version 1.
quit	Exit NTPQ.
passwd	This command prompts you to type in a password (which will not be echoed) which will be 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 is printed as received from the remote server. The only formatting/intepretation done on the data is to transform nonascii data into a printable (but barely understandable) form.
timeout *millseconds*	Specify a timeout period for responses to server queries. The default is about 5000 milliseconds. Note that since NTPQ retries each query once after a timeout, the total waiting time for a timeout will be twice the timeout value set.

Control Message Commands

Each peer known to an NTP server has a 16 bit integer association identifier assigned to it. NTP control messages which carry peer variables must identify the peer the values correspond to by including its association ID. An association ID of 0 is special, and indicates 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 will send a preprogrammed series of messages to obtain the data it needs, and the mreadlist and mreadvar commands, which will iterate over a range of associations.

Table 5 Parameter Description

Parameter	Description
associations	Obtains and prints a list of association identifiers and peer statuses for in-spec peers of the server being queried. The list is printed in columns. The first of these is an index numbering the associations from 1 for internal use, the second the actual association identifier returned by the server and the third the status word for the peer. This is followed by a number of columns containing data decoded from the status word. Note that the data returned by the "associations" command is cached internally in NTPQ. The index is then of use when dealing with stupid servers which use association identifiers which are hard for humans to type, in that for any subsequent commands which require an association identifier as an argument, the form &index may 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 which have a radio clock or other external synchronization will respond positively to this. If the association identifier is omitted or zero the request is for the variables of the "system clock" and will generally get a positive response from all servers with a clock. If the server treats clocks as pseudo-peers, and hence can possibly have more than one clock connected at once, referencing the appropriate peer association ID will show the variables of a particular clock. Omitting the variable list will cause the server to return a default variable display.

Parameter	Description
lassocations	Obtains and prints 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 which retain state for out-of-spec client associations (i.e., fuzzballs). Such associations are normally omitted from the display when the "associations" command is used, but are included in the output of "lassociations".
lpassociations	Print 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 fuzzballs.
lpeers	Like R peers, except a summary of all associations for which the server is maintaining state is printed. This can produce a much longer list of peers from 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.
passociations	Prints association data concerning in-spec peers from the internally cached list of associations. This command performs identically to the "associations" except that it displays the internally stored data rather than making a new query.
peers	Obtains a list of in-spec peers of the server, along with a summary of each peer's state. Summary information includes the address of the remote peer, the reference ID (0.0.0.0 if the refID is unknown), the stratum of the remote peer, the type of the peer (local, unicast, multicast or broadcast), when the last packet was received, the polling interval, in seconds, the reachability register, in octal, and the current estimated delay, offset and dispersion of the peer, all in seconds. The character in the left margin indicates the fate of this peer in the clock selection process. The codes mean: <sp> discarded due to high stratum and/or failed sanity checks; "x" designated falsticker by the intersection algorithm; "." culled from the end of the candidate list; "-" discarded by the clustering algorithm; "+" included in the final selection set; "#" selected for synchronization but distance exceeds maximum; "*" selected for synchronization; and "o" selected for synchronization, PPS signal in use. **NOTE:** Since the peers command depends on the ability to parse the values in the responses it gets it may fail to work from time to time with servers which poorly control the data formats. The contents of the host field may be one of four forms. It may be a host name, an IP address, a reference clock implementation name with its parameter or "REFCLK(<implementation number>, <parameter>)". On "hostnames no" only IP-addresses will be displayed.

Parameter	Description
pstatus *assocID*	Sends a read status request to the server for the given association. The names and values of the peer variables returned will be printed. Note that the status word from the header is displayed preceding the variables, both in hexadecimal and in pidgeon 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 will be those of the corresponding peer. Omitting the variable list will send a request with no data which should induce the server to return a default display.
rvi *index*	Similar to rv with 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.

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 [ -v dn ] [ -r retries ] [ -t timeout ] [ server ]
```

Table 6 NTPTrace Parameters

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

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 this and maybe 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 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 will attempt to read interactive format commands from the standard input.

Table 7 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 will be written to the standard output and commands read from the standard input.

Parameter	Description
-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	Prints a list of the peers known to the server as well as a summary of their state. This is equivalent to **`-c peers`**.
-s	Prints 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`**.

XNTPD

XNTPD is an operating system daemon which sets and maintains the system time-of-day in synchronism 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 a fleet of workstations without specifying configuration details specific to the local environment.

Ordinarily, XNTPD reads the ntp.cfg 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 [ -aAbdm ] [ -c conffile ] [ -f driftfile ] [ -k keyfile ]
[ -l logfile ] [ -p pidfile ] [ -r broadcastdelay ] [ -s statsdir
] [ -t key ] [ -v variable ] [ -V variable ] [-T slp/noncp] [-S]
```

Table 8 XNTPD Parameters

Parameter	Description
-a	Enables authentication mode. The default is enabled, so this option is obsolete now.
-A	Disables authentication mode.
-b	Synchronizes using NTP broadcast messages.
-c *conffile*	Specifies the name and path of the configuration 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.

Parameter	Description
-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.
-m	Synchronizes 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 *variable*	Adds a system variable listed by default.
-T *slp*	-T provides Timesync migration or backward compatibility options. Enables NTP to automatically discover SLP advertising Timesync SINGLE server on the network and add the Timesync SINGLE server's IP address in the ntp.cfg configuration file as a time provider.
-T *noncp*	Stops the ncp engine on XNTPD, which serves all ncp time request's from NetWare 4, Novell clients and dsrepair.
-S	Steps synchronization. XNTPD will set the clock to the time of the best available server and sets the clock status to "nearly in sync". Basically helps to synchronize fast.

The Configuration File

The XNTPD configuration file (ntp.cfg) 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 the -c conffile command line option).

Sample Configuration File

The ntp.cfg looks similar to the following:

```
# sys:\etc\NTP.CFG

#

# 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
# minpoll 4    : Used to synchronize time within a minute. (optional)
#
# server <IP Address> minpoll 4
#
# Peer-Peer Mode
# <IP Address> : Peer IP address
# minpoll 4    : Used to synchronize time within a minute. (optional)
#
# peer <IP Address> minpoll 4
#
# 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.key

# 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 may be optional, separated by whitespace. Commands may not be continued over multiple lines. Arguments may 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. While there is a rich set of options available, the only required option is one or more server, peer or broadcast commands described in the "Configuration Options" on page 38.

- "Configuration Options" on page 38
- "Authentication Options" on page 40
- "Monitoring Options" on page 42

- "Access Control Options" on page 45
- "Miscellaneous Options" on page 46

Files

sys:\etc\ntp.cfg - 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

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 may 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 network(s) or a multicast address assigned to NTP. The Numbers Czar 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 parameter description, see Table 9, "Configuration options parameter description," on page 38.

Table 9 Configuration options parameter description

Parameter	Description
key *key*	All packets sent to the address are to include authentication fields encrypted 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 will be 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. Selection of the proper value, which defaults to 127, is something of a black art and must be coordinated with the network administrator(s).
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 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. Note that, in order to avoid accidental or malicious disruption in this mode, both the local and remote servers should operate 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 address(es) of the global network. The default address is that assigned by the Numbers Czar 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 zero. In this case, it may take some 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 using XNTPDC.

Table 10 Parameter description 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 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 in the multicastclient command with 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 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 11 Stepclock and noncp

Parameter	Description
stepclock	Steps synchronization. XNTPD will set the clock to the time of the best available server and sets the clock status to "nearly in sync". Basically helps to synchronize fast.
noncp	Stops the ncp engine on XNTPD, which serves all ncp time request's from NetWare 4, Novell clients and dsrepair.

Authentication Options

The NTP standard specifies an extension which provides cryptographic authentication of received NTP packets. This is implemented in XNTPD using the DES or MD5 algorithms to compute a digital signature, or message digest. The specification allows any one of possibly 4 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, which should be exchanged and stored 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 for the NTPQ utility program and the third for the XNTPDC utility program.

Authentication Commands

Table 12 Parameter Description

Parameter	Description
keys *keyfile*	Specifies the file name containing the encryption keys and key identifiers used by XNTPD, NTPQ and XNTPDC when operating in authenticated mode. The format of this file is described later in this document.
trustedkey *key* [...]	Specifies the encryption key identifiers which 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. Note that 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 will be 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 will be ignored.

Authentication Key File Format

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

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

Table 13 Parameter Description

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 such that odd parity is maintained for the octet. Leading zeroes must be specified (i.e., the key must be exactly 16 hex digits long) and odd parity must be maintained. Hence 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. Note that both the keys and the authentication schemes (DES or MD5) must be identical between a set of peers sharing the same key number.

Note that 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.

Monitoring Options

XNTPD includes a comprehensive monitoring facility suitable for continuous, long term recording of server and client timekeeping performance. See the statistics command below for a listing and example of each type of statistics currently supported. Statistic files are managed 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.

Monitoring Commands

Table 14 Parameter Description

Parameter	Description
statistics *name* [...]	Enables writing of statistics records. Currently, three kinds of name statistics are supported.

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 status, respectively. The status field is encoded in hex in the 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 a good deal of additional information can be gathered and displayed as well. 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 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 at most one element of the set is being written to. The type given specifies when and how data will be directed to a new element of the set. This way, information stored in elements of a file set that are currently unused are available for administrational operations without the risk of disturbing the operation of XNTPD. (Most important: 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 may, however, 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 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 is 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 a 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 file name suffix consists of a dot, a 4-digit year number, and a 2-digit month.

Parameter	Description
	◆ **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 using nolink. If link is specified, a hard link from the current file set element to a file without 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.

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. While this facility may 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.

Access Control Commands

Table 15 Parameter Description

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 may 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, though 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 no longer is counted for client limit restriction. The default value for this variable is 3600 seconds.

Miscellaneous Options

Miscellaneous Commands

Table 16 Parameter Description

Parameter	Description
broadcastdelay *seconds*	The broadcast and multicast modes require a special calibration to determine the network delay between the local and remote servers. Ordinarily, this is done automatically by the initial protocol exchanges between the local and remote servers. In some cases, the calibration procedure may fail due to network or server access controls, for example. This command specifies the default delay to be used under these circumstances. Typically (for Ethernet), a number between 0.003 and 0.007 seconds is appropriate. The default when this command is not used is 0.004 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 may 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 will be 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 that they can be listed. The known protocol variables will always override any variables defined via the setvar mechanism. There are three special variables that contain the names of all variable 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.

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 synchronizations source is ignored (thus LEAP_NOWARNING is passed on).

5 Monitoring and Security

This chapter describes NTP monitoring and security.

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.

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

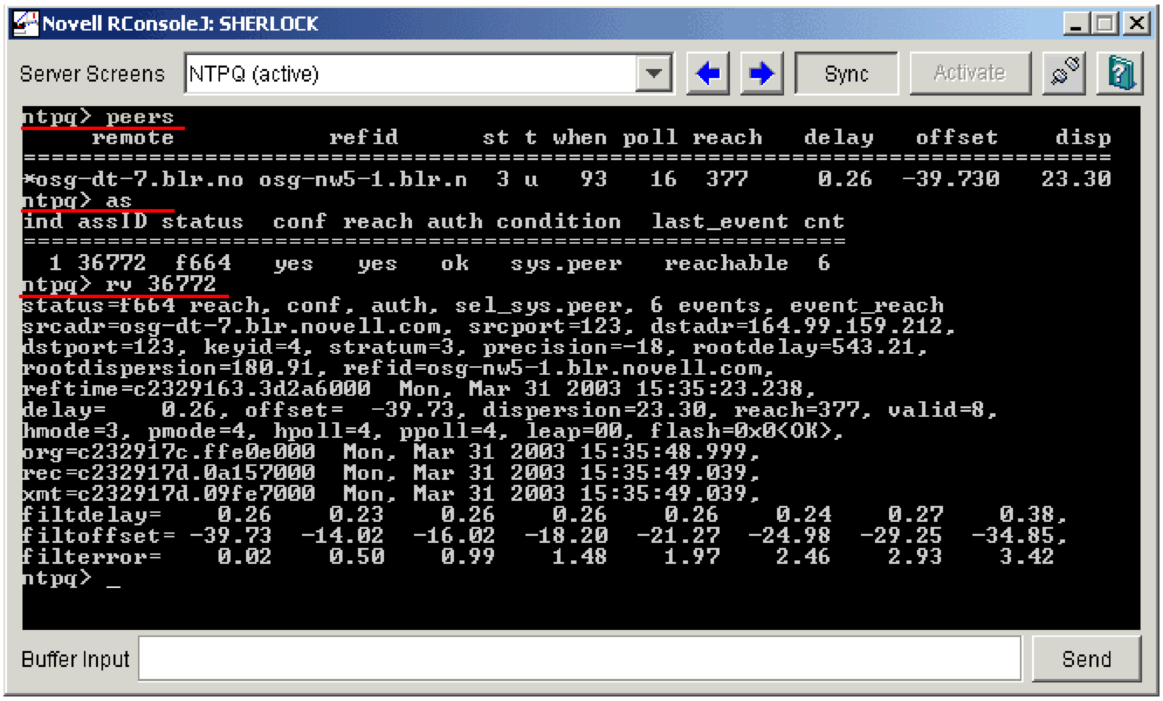
Table 17 NTPQ commands

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 prints 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).
re *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 “NTPQ” on page 28.

The following figure display the output of the NTPQ peers, association and rv *assocID* commands.

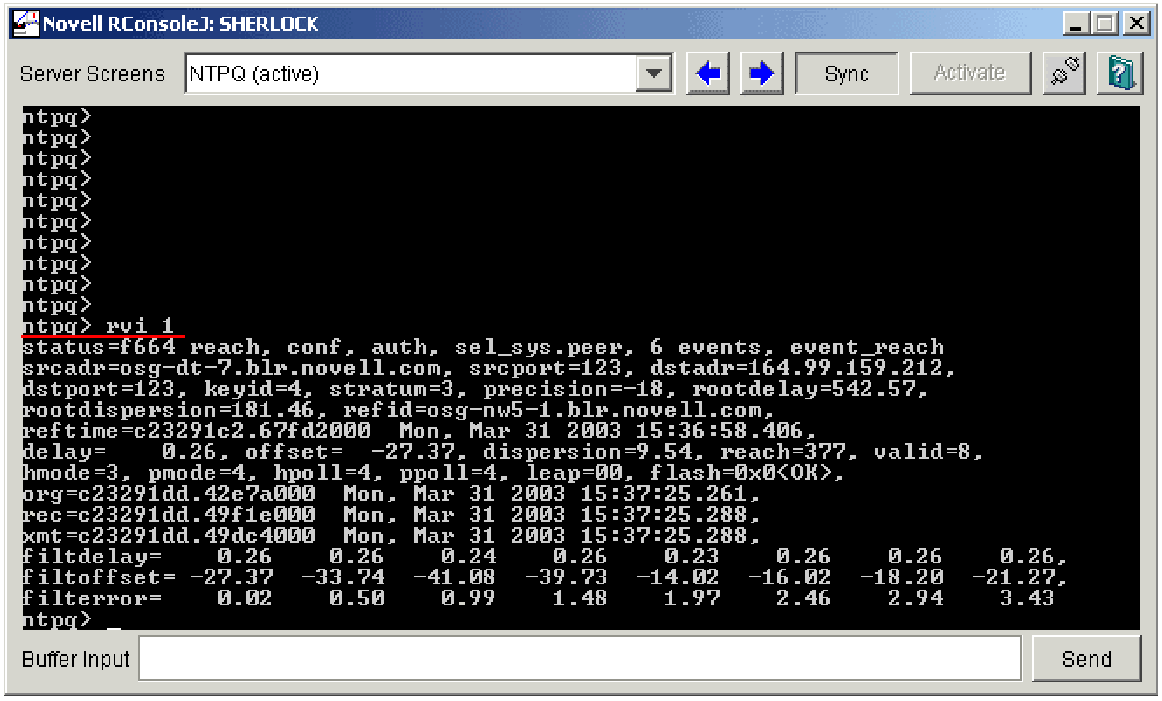
Figure 11 Output of NTPQ peers, association and rv *assocID* commands



```
ntpq> peers
     remote           refid      st t when poll reach   delay   offset    disp
==============================================================================
*osg-dt-7.blr.no osg-nw5-1.blr.n  3 u   93   16  377     0.26  -39.730   23.30
ntpq> as
ind assID status  conf reach auth condition  last_event cnt
===========================================================
  1 36772  f664   yes   yes   ok   sys.peer   reachable  6
ntpq> rv 36772
status=f664 reach, conf, auth, sel_sys.peer, 6 events, event_reach
srcadr=osg-dt-7.blr.novell.com, srcport=123, dstadr=164.99.159.212,
dstport=123, keyid=4, stratum=3, precision=-18, rootdelay=543.21,
rootdispersion=180.91, refid=osg-nw5-1.blr.novell.com,
reftime=c2329163.3d2a6000  Mon, Mar 31 2003 15:35:23.238,
delay=    0.26, offset=  -39.73, dispersion=23.30, reach=377, valid=8,
hmode=3, pmode=4, hpoll=4, ppoll=4, leap=00, flash=0x0<OK>,
org=c232917c.ffe0e000  Mon, Mar 31 2003 15:35:48.999,
rec=c232917d.0a157000  Mon, Mar 31 2003 15:35:49.039,
xmt=c232917d.09fe7000  Mon, Mar 31 2003 15:35:49.039,
filtdelay=    0.26    0.23    0.26    0.26    0.26    0.24    0.27    0.38,
filtoffset= -39.73  -14.02  -16.02  -18.20  -21.27  -24.98  -29.25  -34.85,
filterror=    0.02    0.50    0.99    1.48    1.97    2.46    2.93    3.42
ntpq> _
```

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

Figure 12 Output of the NTPQ rvi *index* command

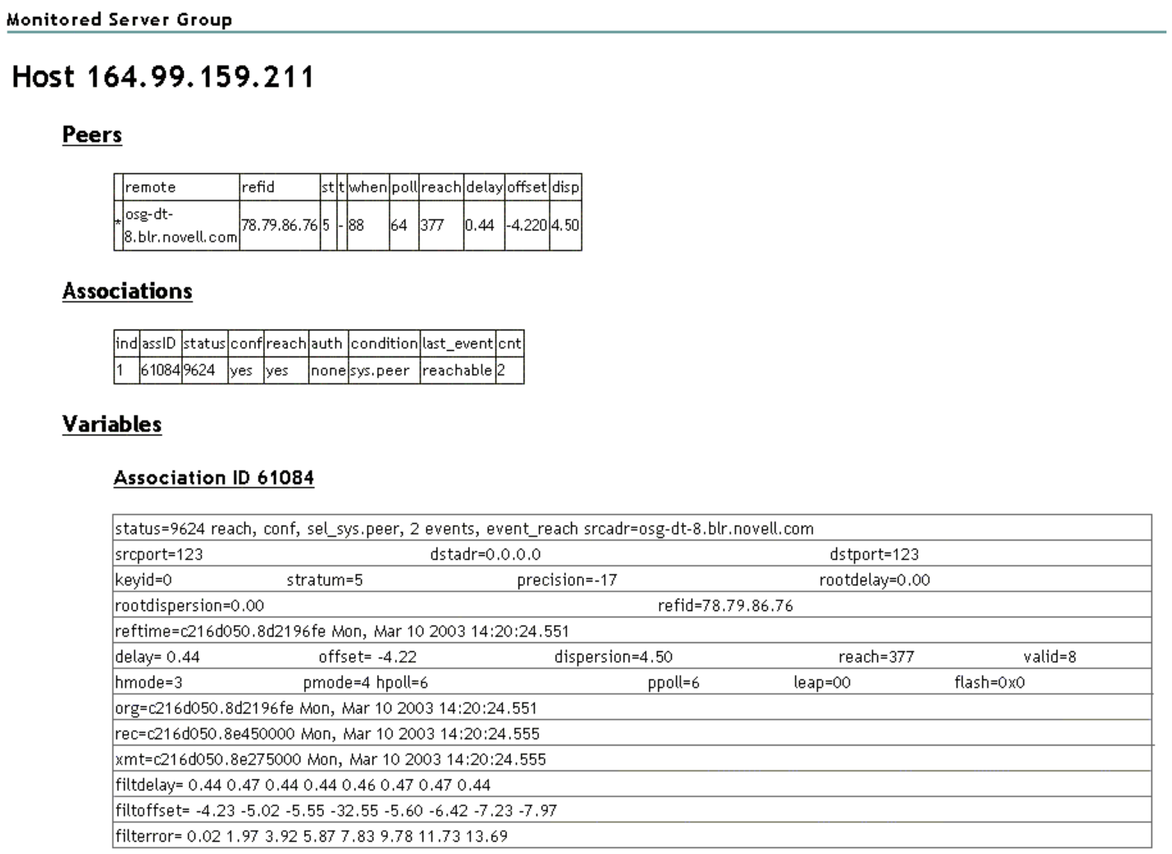


```
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq>
ntpq> rvi 1
status=f664 reach, conf, auth, sel_sys.peer, 6 events, event_reach
srcadr=osg-dt-7.blr.novell.com, srcport=123, dstadr=164.99.159.212,
dstport=123, keyid=4, stratum=3, precision=-18, rootdelay=542.57,
rootdispersion=181.46, refid=osg-nw5-1.blr.novell.com,
reftime=c23291c2.67fd2000  Mon, Mar 31 2003 15:36:58.406,
delay=    0.26, offset=  -27.37, dispersion=9.54, reach=377, valid=8,
hmode=3, pmode=4, hpoll=4, ppoll=4, leap=00, flash=0x0<OK>,
org=c23291dd.42e7a000  Mon, Mar 31 2003 15:37:25.261,
rec=c23291dd.49f1e000  Mon, Mar 31 2003 15:37:25.288,
xmt=c23291dd.49dc4000  Mon, Mar 31 2003 15:37:25.288,
filtdelay=    0.26    0.26    0.24    0.26    0.23    0.26    0.26    0.26,
filtoffset= -27.37  -33.74  -41.08  -39.73  -14.02  -16.02  -18.20  -21.27,
filterror=    0.02    0.50    0.99    1.48    1.97    2.46    2.94    3.43
ntpq> _
```

Health Monitor

You can monitor NTP remotely using NORM. You can monitor all the servers that are authenticated to the eDirectory™ tree. Select the server you want to monitor. The Peer and Associations details are displayed. These details are similar to output the NTPQ peers and NTPQ associations command gives. For more information, see Table 17, “NTPQ commands,” on page 49.

Figure 13 Monitoring NTP using NORM



Monitored Server Group

Host 164.99.159.211

Peers

	remote	refid	st	t	when	poll	reach	delay	offset	disp
*	osg-dt-8.blr.novell.com	78.79.86.76	5	-	88	64	377	0.44	-4.220	4.50

Associations

ind	assID	status	conf	reach	auth	condition	last_event	cnt
1	61084	9624	yes	yes	none	sys.peer	reachable	2

Variables

Association ID 61084

status=9624 reach, conf, sel_sys.peer, 2 events, event_reach srcadr=osg-dt-8.blr.novell.com
srcport=123 dstadr=0.0.0.0 dstport=123
keyid=0 stratum=5 precision=-17 rootdelay=0.00
rootdispersion=0.00 refid=78.79.86.76
reftime=c216d050.8d2196fe Mon, Mar 10 2003 14:20:24.551
delay= 0.44 offset= -4.22 dispersion=4.50 reach=377 valid=8
hmode=3 pmode=4 hpoll=6 ppoll=6 leap=00 flash=0x0
org=c216d050.8d2196fe Mon, Mar 10 2003 14:20:24.551
rec=c216d050.8e450000 Mon, Mar 10 2003 14:20:24.555
xmt=c216d050.8e275000 Mon, Mar 10 2003 14:20:24.555
filtdelay= 0.44 0.47 0.44 0.44 0.46 0.47 0.47 0.44
filtoffset= -4.23 -5.02 -5.55 -32.55 -5.60 -6.42 -7.23 -7.97
filterror= 0.02 1.97 3.92 5.87 7.83 9.78 11.73 13.69

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, while 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, furthermore, 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, which the local server is unable to decrypt
- Peers that send authenticated packets encrypted using a key NTP does not trust

NOTE: Though 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.key    # path for key file

trustedkey 1 2 14 15     # define trusted keys

requestkey 15            # key (7) for accessing server variables

controlkey 15            # key (6) for accessing server variables

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

- The `enable auth` line enables authentication processing.
- The `keys sys:\etc\ntp.key` 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.key 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.
- The `authdelay` line is an estimate of the amount of processing time taken between the freezing of a transmit time stamp and the actual transmission of the packet when authentication is enabled (generally, the time it takes for the DES or MD5 routine to encrypt

a single block). Additionally, it is used as a correction for the transmit timestamp. This can be computed for your CPU by the `authspeed` program.

The usage of DES and MD5 keys is illustrated by the following:

```
# for DES keys

authspeed -n 30000 auth.samplekeys

# for MD5 keys

authspeed -mn 30000 auth.samplekeys
```

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.

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

WARNING: This file is better left unreadable by anyone except admin.

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

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

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 18 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.

Key Format	Description
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: Due to 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.

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.

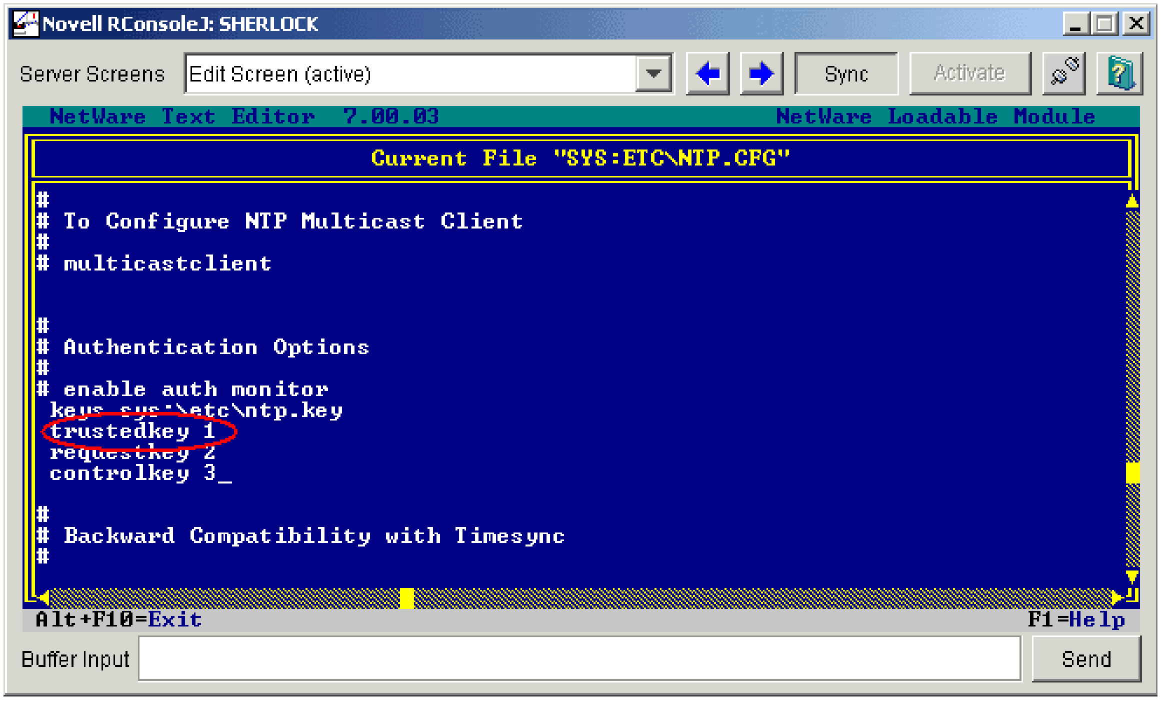
On ServerA:

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

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

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

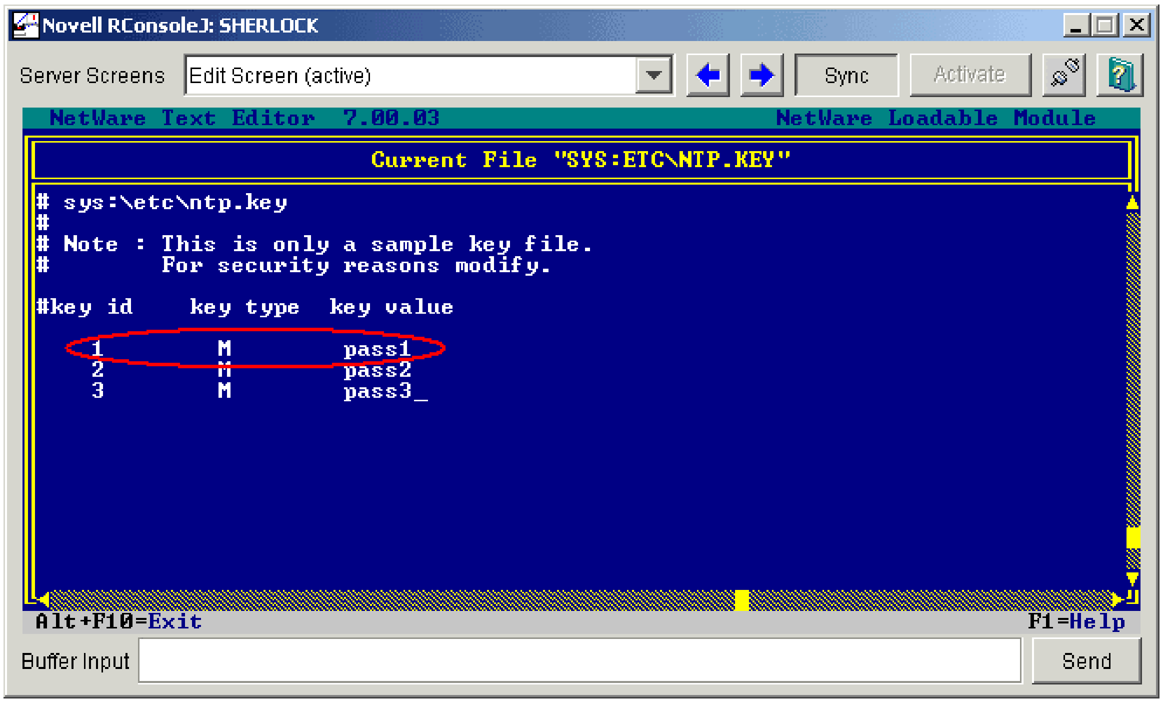
Figure 14 **ntp.cfg file of ServerA**



```
NetWare Text Editor  7.00.03                    NetWare Loadable Module
                   Current File "SYS:ETC\NTP.CFG"
#
# To Configure NTP Multicast Client
#
# multicastclient


#
# Authentication Options
#
# enable auth monitor
 keys sys:\etc\ntp.key
 trustedkey 1
 requestkey 2
 controlkey 3_

#
# Backward Compatibility with Timesync
#
Alt+F10=Exit                                                    F1=Help
```

- In the ntp.key file located in sys:\etc, give a key value for key ID 1, pass1.

Figure 15 ntp.key file of ServerA



Current File "SYS:ETC\NTP.KEY"

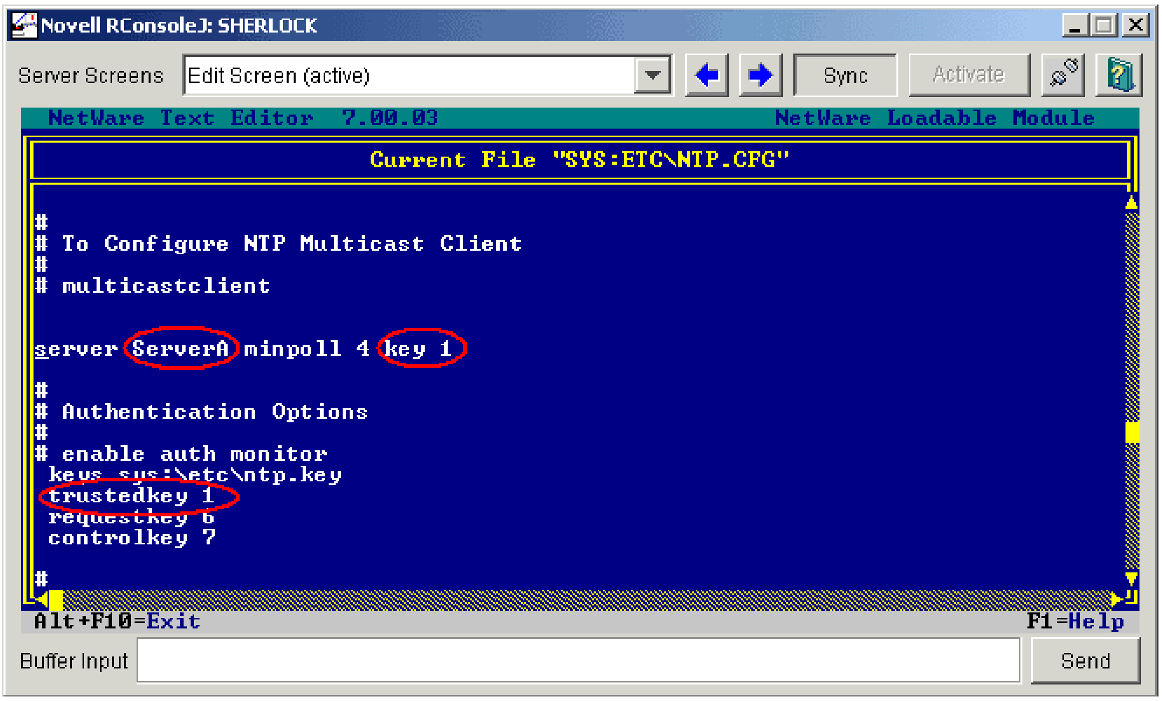
```
# sys:\etc\ntp.key
#
# Note : This is only a sample key file.
#        For security reasons modify.

#key id    key type   key value

    1          M          pass1
    2          M          pass2
    3          M          pass3_
```

On ServerB:

- In the ntp.cfg 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.key`
  - Mark 1 as the trusted key ID. See the following figure for more details.

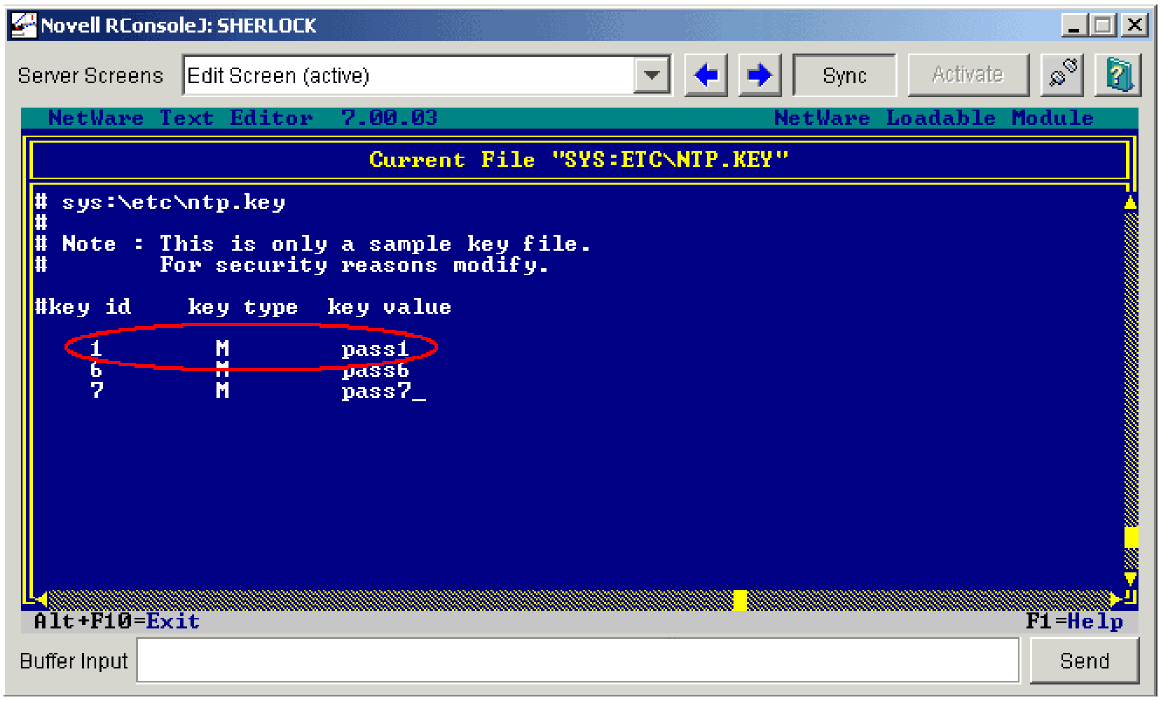
Figure 16 ntp.cfg file of ServerB



```
NetWare Text Editor  7.00.03                    NetWare Loadable Module
                 Current File "SYS:ETC\NTP.CFG"

#
# To Configure NTP Multicast Client
#
# multicastclient

server ServerA minpoll 4 key 1

#
# Authentication Options
#
# enable auth monitor
 keys sys:\etc\ntp.key
 trustedkey 1
 requestkey 6
 controlkey 7

#
Alt+F10=Exit                                                    F1=Help
```

- In the ntp.key 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 17 ntp.key file of ServerB



```
NetWare Text Editor  7.00.03                    NetWare Loadable Module
                 Current File "SYS:ETC\NTP.KEY"
# sys:\etc\ntp.key
#
# Note : This is only a sample key file.
#        For security reasons modify.

#key id    key type  key value

    1         M         pass1
    6         M         pass6
    7         M         pass7_

Alt+F10=Exit                                                    F1=Help
```

After entering the details in ServerA and ServerB's ntp.cfg and ntp.key files, restart XNTPD on both the servers.

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 talk only NCP™. NetWare 5.*x* and 6.*x* can talk NCP and NTP.

NTPv3 components do not support NCP (backward compatibility for Timesync).

6 Troubleshooting NTP

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

- "XNTPD" on page 59
- "NTPDate" on page 61
- "General" on page 61

XNTPD

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

XNTPD -T slp does not update ntp.cfg

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`***.

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.

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.cfg file and the request key from the ntp.key 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.key file, and mark it as a trusted key in ntp.cfg. Ensure that this key is present in the broadcast/multicast server.

XNTPD takes a long time to synchronize

Problem: The polling interval is large.

Action: Minimize the polling interval using minpoll. Set minpoll equal to 4 in the ntp.cfg file while giving the time provider details.

Action: Use XNTPD with the -S option or stepclock in the ntp.cfg file to synchronize within 10 seconds.

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 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 does not get 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 the 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 be achieved only through authentication.

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

Action: Obtain the time provider's key, copy it to the ntp.key file, and mark it as a trusted key in the ntp.cfg 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 broadcasted or multicasted.

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

```
server 127.127.0.1 prefer
```

XNTPD hangs while loading

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

Time synchronization 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 large; therefore, XNTPD does not adjust the clock and it exits.

Action: Start XNTPD with the -S option.

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 is for the error details.

General

This section provides solutions to generic problems, you might encounter.

Unable to configure NTPv3 as time client

Action: Check if the time provider is broadcasting/multicasting its service.

Action: Check if time sources are in synchronization.

Action: In case 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.

7 Frequently Asked Questions

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

How do I configure NTP for fault tolerance?

Configure two servers as follows.

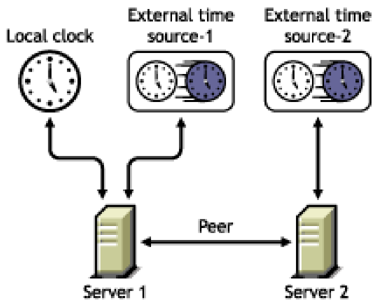
For Server1:

- 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.

For Server2:

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

Figure 18 Fault Tolerance Configuration



Server1 and Server2 will provide time to all other servers in the network.

How do I find the subnet broadcast address?

Start XNTPD with debug level 4 and log option enabled. XNTPD displays the subnet broadcast address for each time source.

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 (*).

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 out the offset.
- Run NTPQ with the client and enter **`pe`**. This lists the offset of the client with each time source.