

# Network Time Protocol (NTP)

## Feature Overview and Configuration Guide

### Introduction

NTP is a protocol designed to synchronize the clocks of computers over a network. The objective of NTP is simple: to allow a client to synchronize its clock with Coordinated Universal Time (UTC), and to do so with a high degree of accuracy and stability.

UTC time is disseminated by various means to reference clocks that accurately provide the current time. Typically reference clocks are highly precise atomic clocks and GPS clocks.

Thousands of NTP servers around the world have access to the reference clocks, and act as primary time servers. These then synchronize a much larger number of secondary servers and clients connected by a common network.

NTP protocol mechanisms specify the precision and estimated error of the local clock and the characteristics of the reference clock to which it may be synchronized.

NTP belongs to and is one of the oldest parts of the TCP/IP protocol suite. NTP applies to both the protocol and the client-server implementation that run on computers.

Understanding NTP is important for network administrators managing, securing, planning, and debugging a network, as time is often a critical factor in mapping events together.

#### Guide content

This guide begins with an overview of the NTP protocol and its basic concepts such as clock strata levels and NTP timestamps. It continues with a description of its operation and the roles and modes of operation. A couple of configuration examples are provided: the first example describes how to configure two switches, one at a Head Office and one at a Regional Office, to provide a network time service. The second example describes how to setup NTP authentication.

This guide concludes with a short troubleshooting section.

## Products and software version that apply to this guide

This guide applies to all AlliedWare Plus™ products, running version **5.4.8-1.1** or later. Feature support may change in later software versions. For the latest information, see the following documents:

- The [product's Datasheet](#)
- The [AlliedWare Plus Datasheet](#)
- The product's [Command Reference](#)

These documents are available from the above links on our website at [alliedtelesis.com](http://alliedtelesis.com).

**Note:** Some of the NTP background information in this guide is sourced from the [Network Time Synchronization Project](#). It does not describe the architecture, protocols and algorithm in full technical detail.

# Contents

Introduction .....	1
Products and software version that apply to this guide .....	2
NTP Overview.....	4
Why is NTP so important in a computer network? .....	4
NTP Basic Concepts.....	5
Clock strata levels and the reference clock .....	5
NTP Timestamps .....	7
NTP timekeeping metrics.....	8
Servers and clients .....	8
Operation of the NTP Protocol.....	9
Roles and Modes of Operation .....	9
NTP on AlliedWare Plus Devices.....	10
Is the Device Clock Synchronized to the NTP Server? .....	11
Understanding the 'show ntp associations' command output.....	11
Understanding the 'show ntp status' command output.....	14
Losing communication with the clock server .....	15
How to configure NTP.....	15
Configuring NTP in the Device GUI .....	16
Configuring NTP in the CLI .....	17
NTP Authentication .....	19
Configuring authentication on the NTP client.....	19
Configuring authentication on the NTP server.....	20
Authentication and filtering for NTP peers.....	21
Troubleshooting.....	22

## NTP Overview

NTP is used to synchronize device clocks in a packet-switched, variable latency network to within a few milliseconds of Coordinated Universal Time (UTC). In fact, NTP provides nominal accuracies of low tens of milliseconds on WANs, sub-milliseconds on LANs, and sub-microseconds using a precision time source such as a cesium oscillator or GPS receiver.

NTP software has been ported to almost every workstation and server platform available today - from PCs to Unix, Windows, OpenVMS and embedded systems, even home routers.

NTP is one of the oldest Internet protocols in current use having been in operation since before 1985. NTP is instrumental in synchronizing the clocks on an estimated 25 million servers, workstations and PCs of the public Internet and private networks. Both peer-to-peer and client-server implementations use NTP to send and receive timestamps over UDP.

NTP was originally designed by David L. Mills of the University of Delaware. In 1985, NTP version 0 was implemented and documented in RFC 958. The protocol has been developed considerably and NTP version 4 is the current development version documented in RFC 5905.

### Why is NTP so important in a computer network?

For network administrators, NTP is important because managing, securing, planning, and debugging a network involves determining when events happen.

Accurate time on network devices is required for: Digital certificate validation, logging with accurate timestamps, and time-based traffic restrictions, such as those configured in access control lists (ACLs).

Accurate timestamping is key to root-cause analysis, determining when problems occurred and finding correlations. If network devices are out of sync by a few milliseconds, or in extreme cases a few seconds, it can be very difficult to determine the sequence of events.

NTP is also important for:

- Tracking security breaches, network usage, or problems affecting a large number of components.
- Reducing confusion in shared file systems. It is important file modification times to be consistent, regardless of what machine the file systems are on.
- Billing services and similar applications that must know the time accurately.
- Timekeeping - some legislation and financial services require highly accurate timekeeping by law.

## NTP Basic Concepts

In its simplest form, the NTP protocol is a clock request transaction. The objective being to allow a client to synchronize its clock with an accurate UTC reference source. UTC is based on International Atomic Time (TAI), which is derived from hundreds of cesium and hydrogen clocks in the national standards laboratories of many countries.

### What does NTP do?

NTP synchronizes the clocks of computer systems. It can read the time from a reference source, then transmit the reading to one or more clients and adjust each client clock as required. NTP servers listen for client NTP packets on port 123. The NTP server is stateless and responds to each received client NTP packet in a simple transactional manner by adding fields such as timestamps to the received packet and passing the packet back to the original sender, without reference to preceding NTP transactions.

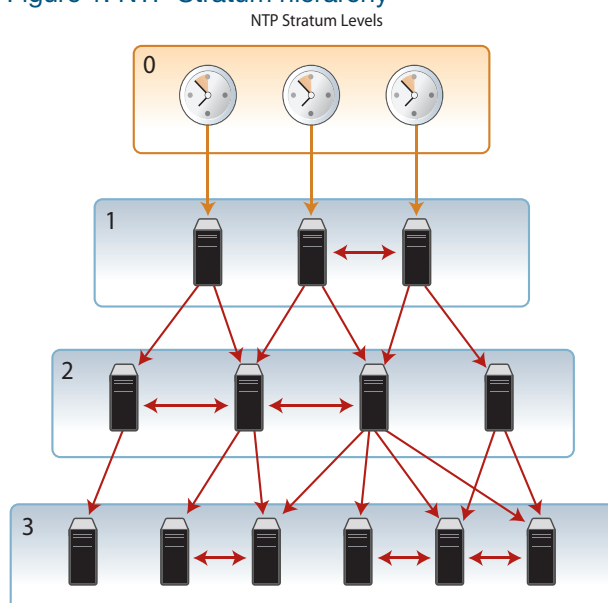
Let's look at some NTP concepts in more detail: Clock strata levels and the reference clock, NTP Timestamps, timekeeping, and servers and clients.

### Clock strata levels and the reference clock

NTP is a hierarchical protocol and is divided into stratum which define the **distance** from the **reference clock** of time sources. A reference clock is the device that sits atop the stratum hierarchy and is typically a cesium atomic or a GPS clock. A stratum is simply one layer (or hop count) in the network of layers that distribute time across a network of devices. Looking at [Figure 1 on page 5](#), you can see that the top stratum level is 0. This is where the most accurate reference clock sources are located.

Then going down the diagram there are stratum1 devices obtaining their time from the stratum 0 level. Then stratum 2 devices obtain their time from stratum 3, and so on down. The further down the diagram (the higher stratum number) the less accurate the clock time potentially becomes.

Figure 1: NTP Stratum hierarchy



Here is a brief description of the stratum levels:

### Stratum 0

- Stratum 0 time sources are very high precision clocks such as atomic clocks that have little or no delay associated with them. They are known as primary **reference** clocks, and synchronized to national standards such as UTC.
- Stratum 0 devices cannot be used on the network, instead, they are directly connected to computers (such as backbone gateways or switches) which then operate as primary time servers, or Stratum 1.

### Stratum 1

- Stratum 1 devices (primary time servers), have their system time synchronized to within a few microseconds of their attached stratum 0 devices.
- Stratum 1 time servers use NTP between them to cross check clocks, to mitigate errors due to equipment or propagation failures, and to distribute time information to local secondary time servers (stratum 2).

### Stratum 2

- Stratum 2 devices synchronize their system time via NTP packet request from stratum 1 (primary time servers). Often a stratum 2 computer will query several stratum 1 servers.
- Stratum 2 devices can be used as a time source and as equipment that connects to other stratum 2 devices.

### Stratum 3 and on

- Stratum 3 devices are synchronized to stratum 2 servers. They employ the same algorithms for peering and data sampling as stratum 2, and can themselves act as servers for stratum 4 computers, and so on.

### How many stratum levels are there?

The upper limit for stratum is 15; stratum 16 is used to indicate that a device is unsynchronized.

Multiple stratum are used in larger networks because to bombard a single stratum 1 time server with NTP requests from thousands of machines could cause it to overload or block the network itself with workstations/routers etc. repeatedly waiting for their time synchronization requests to go through.

It is usually necessary at each stratum level to employ redundant servers and diverse network paths in order to protect against broken software, hardware, networks and potential hostile attack.

### What stratum level is typically used in organizations?

As you progress through different strata there are network delays involved that reduce the accuracy of the NTP server in relation to UTC.

Typically, the NTP process in an individual organization's network will sync to a stratum 3 or stratum 4 time source, and use that source to provide a reference time for the devices in the network to synchronize to.

To make the system more reliable, each client can receive a time source from multiple servers. Stratum 2 devices and below can also synchronize with each other. NTP continuously monitors the stability and accuracy of all the servers and always chooses the most accurate one.

## NTP Timestamps

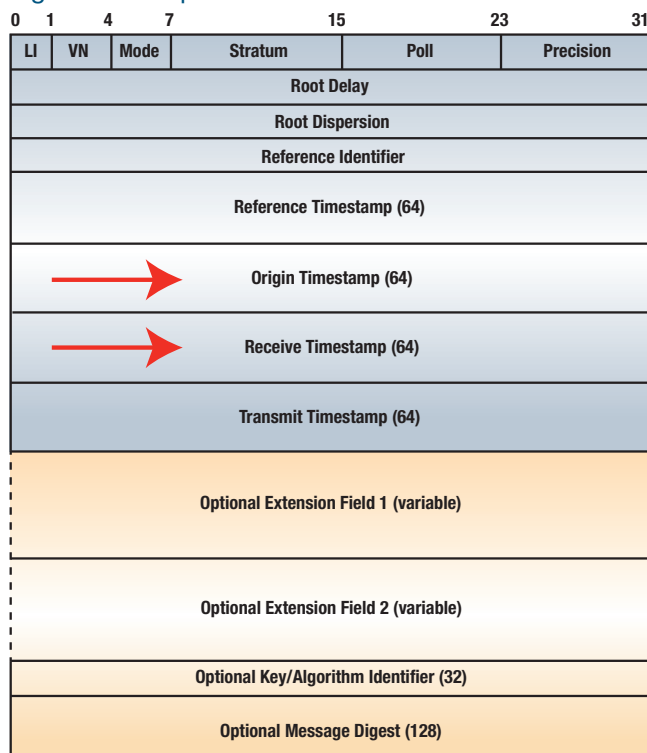
As mentioned earlier, the objective of NTP is simple: to allow a client to accurately synchronize its clock with Coordinated Universal Time. Synchronizing a client to a network server consists of several NTP packet exchanges where each exchange is a pair of request and reply.

- When sending out a request, the client stores its own time (the origin timestamp) into the packet being sent. When a server receives such a packet, it will in turn store its own time (receive timestamp) into the packet, and the packet will be returned after putting a transmit timestamp into the packet.
- When receiving the reply, the receiver will once more log its own receipt time to estimate the travelling time of the packet. The travelling time (delay) is estimated to be half of "the total delay minus remote processing time", assuming symmetrical delays.

The NTP packets sent by the client to the server and the responses from the server to the client use a common format. There are two NTP time formats, a 64-bit timestamp and a 128-bit datestamp. The datestamp format is used internally, while the timestamp format is used in packet header exchanges between clients and servers.

You can see the origin and receive timestamps in the NTP packet below:

Figure 2: NTP packet structure



## NTP timekeeping metrics

NTP is designed to allow a computer to be aware of three critical metrics for timekeeping:

- the **offset** (or phase) of the local clock to a selected reference clock. The offset is the clock time difference between the peers or between the reference and local clock. This value is the correction that is applied to a client clock in order to synchronize it.
- the round-trip **delay** of the network path between the local computer and the selected reference clock server.
- the **jitter** (or dispersion) of the local clock, which is a measure of the maximum error of the local clock relative to the reference clock.

These metrics are maintained separately in NTP, and provide definitive maximum error boundaries for the synchronization process. This means that you can determine both the accuracy and quality of time in a network.

## Servers and clients

An NTP server is a source of time information, and an NTP client is a system/device that is attempting to synchronize its clock to a server.

**Servers** can be either a primary or secondary server:

- A primary server receives UTC time signals directly from a very accurate source such as an atomic clock or more commonly - a GPS signal source.
- A secondary server receives its time signal from one or more upstream servers, and distributes its time signals to one or more downstream servers and clients. Secondary servers are arranged in a strict hierarchy in terms of upstream and downstream, and the stratum terminology is often used to assist in this process.

**Clients** peer with servers in order to synchronize their internal clocks to the NTP time signal.



## Operation of the NTP Protocol

NTP operates over the User Datagram Protocol (UDP). As mentioned earlier, NTP servers listen for client NTP packets on port 123. The NTP server is stateless and responds to each received client NTP packet in a simple transactional manner by adding fields to the received packet and passing the packet back to the original sender, without reference to preceding NTP transactions.

In brief, the process that occurs in NTP is:

1. Devices send out timestamp requests.
2. Other devices receive the requests and respond with packets containing timestamps, indicating their current system time. See "[NTP packet structure](#)" on page 7.
3. The requesting devices receive these timestamps. They do not simply set their system time to the exact value contained in the received timestamp. They need to allow for the time it took the packet to travel to them. So, they need to calculate the travel time from the responder to themselves.
4. The travel time is calculated by knowledge of the time the request was sent, and when the response arrived, in addition to the time that the responder believed it received the request and sent its response. These values are all mashed together into formulae, out of which pop the travel time of the packets, and the time difference between the two devices' system clocks.
5. Repeated iterations of this procedure allow the local client to remove the effects of network jitter and thereby gain a stable value for the delay between the local clock and the reference clock standard at the server.
6. The polling process is designed to provide a sufficient update rate to maximize accuracy while minimizing network overhead.

## Roles and Modes of Operation

In the protocol description above, it is evident that there are different roles that devices play in the process. There's the requester role and the responder role.

There are two fundamental **modes** in which the NTP exchange can operate:

- **Client/Server mode:** The client requests and the server responds. The server will be a higher-stratum (i.e., lower stratum number) device than the client. So, the client's time converges onto the server's time.
- **Peer-to-peer mode:** Both devices in the conversation act as requesters and responders to each other. The peer devices are both at the same stratum. The devices' times converge to each other.

Of course, a device that is operating as a client to a higher-stratum server can, at the same time, also be operating as a server to other, lower-stratum clients.

Moreover, a device that is operating as client or server to some partners, can, at the same time, also be operating as a peer to yet others.

The roles of client server and peer do not define the operation of a whole NTP device, they simply describe the relationship with another specific NTP node conversation in which it might be partaking at any given time.

In summary, an NTP device can take on one or more of the following three roles:

- **Client**— it can be a client to one or more servers, from whom it requests reference timestamps.
- **Server**—it sends reference timestamps in response to requests from one or more clients.
- **Peer**—it exchanges timestamps with other peers until all the peers agree on what the real time is. The agreed upon time is then used as the time to synchronize to.

## NTP on AlliedWare Plus Devices

The implementation of NTP is based on the following RFCs:

- RFC 958, Network Time Protocol (NTP)
- RFC 5905, Network Time Protocol (Version 4) Specification, Implementation and Analysis
- RFC 1510, The Kerberos Network Authentication Service (V5)

AlliedWare Plus supports client, server, and peer modes. Devices can act in all three roles simultaneously.

If a switch receives a client request it then acts as a server to respond to the client, but it will itself still be acting as a client to its configured server(s).

When the device is operating in client mode, then the server(s) to which it refers to must be more accurate clock source(s) than itself or another device directly connected to a more accurate clock source.

If the device receives a synchronization request from an NTP client, it replies to the request with the current time from the device's internal clock along with other information useful for synchronization. The device's internal clock is accurate to 0.005 seconds.

## Is the Device Clock Synchronized to the NTP Server?

The best way to ascertain if the device clock is synchronized to the NTP server is to use the **show ntp associations** and **show ntp status** commands.

- **show ntp associations** - in the output, an asterisk "\*" next to the remote address, indicates the **last** server it was synced to. The other columns in the associations table show when the last contact with a server was, even if the device is not currently synced with that server.
- **show ntp status** - shows the details of the last server synced with,

In the **show ntp associations** output below, the local device was last synced with the server 192.168.1.1. You can see that the server sent time information 3 seconds ago ('when'), and the device polls the server every 64 seconds:

```
awplus#show ntp associations
remote          refid          st  t  when poll reach  delay  offset  disp
-----
*192.168.1.1    202.46.177.18  2   u   3   64   17   0.144  0.006  0.001
```

The next section provides more detail on the **show ntp associations** output and also a couple of examples to explain what happens if server connection is lost or when there is more than one reachable eligible server. This is followed by a brief look at the **show ntp status** command output.

### Understanding the 'show ntp associations' command output

The output of the **show ntp associations** command has a column called 'when'. This shows the number of seconds since the last time the local device received time information from that server. If the local device hasn't synced with that server yet, there will just be "-" in that column.

When it has synced, it will show how long since it last synced. The longer the number of seconds since the last sync, the more likely the local device time has drifted out of sync with the time server. But typically the local device only polls the server every 64 - 128 secs (and not more often than every 64 seconds after initial syncing), and the local time daemon varies the poll rate automatically based on network conditions.

The poll rate can be anything from 64 seconds to every 1024 seconds. The poll rate can gradually increase if the server doesn't respond to polls, however the local time daemon automatically manages the poll rate based on several things including network conditions. So a high poll rate should not be used by itself as an indicator of connectivity problems.

**Example 1 - lost server connection**

In the example below, we have a DUT with two NTP servers:

- ntp server 192.168.2.1 - server 1
- ntp server 192.168.1.1 - server 2

The output of the command **show ntp associations** shows a working link to server 1 and a broken link to server 2:

```
awplus#show ntp associations
remote          refid          st t when poll reach  delay  offset  disp
-----
*192.168.2.1    202.46.185.18  2 u  1  64   17   0.157  0.007  0.002 <==server 1
192.168.1.1    -              0 u  -  64   0     -      -      -      <==server 2
* system peer, # backup, + candidate, - outlier, x false ticker
```

If the link to server 1 is then broken and server 2 is reconnected, then after awhile this is what we see in the command output:

```
awplus#show ntp associations
remote          refid          st t when poll reach  delay  offset  disp
-----
*192.168.2.1    202.46.185.18  2 u 169  64  376   0.182  0.002  0.001 <==server 1
192.168.1.1    202.46.185.18  2 u  35 256   1   0.243  0.094  0.002 <==server 2
* system peer, # backup, + candidate, - outlier, x false ticker
```

The connection to server 1 has not timed out yet, so the DUT is still synced to it, even though it has now contacted server 2.

The output from **show ntp status** confirms this:

```
awplus#show ntp status
Reference ID    : COA80201 (192.168.2.1) <=== Note still referencing server 1
Stratum        : 3
Ref time (UTC) : Mon Nov 26 23:38:00 2018
System time    : 0.000000210 seconds fast of NTP time
Last offset    : +0.000000073 seconds
RMS offset     : 0.000004411 seconds
Frequency      : 17.124 ppm slow
Residual freq  : +0.002 ppm
Skew           : 0.175 ppm
Precision      : -20 (0.000000954 seconds)
Root delay     : 0.020420199 seconds
Root dispersion: 0.001664547 seconds
Update interval: 64.9 seconds
Leap status    : Normal
```

After about 518 seconds since server 1 responded to the DUT (shown in the 'when' field), the designations of the servers starts to change:

```
awplus#show ntp associations
remote      refid          st t when poll reach  delay  offset  disp
-----
*192.168.2.1 202.46.185.18 2 u  518  64  200  0.182  0.002  0.001 <==server 1
192.168.1.1  202.46.185.18 2 u  126 128   3  0.170  0.024  0.002 <==server 2
* system peer, # backup, + candidate, - outlier, x false ticker

awplus#show ntp associations
remote      refid          st t when poll reach  delay  offset  disp
-----
*192.168.2.1 202.46.185.18 2 u  522  64  200  0.182  0.002  0.001 <==server 1
-192.168.1.1 202.46.185.18 2 u   0 128   7  0.179  0.034  0.002 <==server 2
* system peer, # backup, + candidate, - outlier, x false ticker
```

Note the '-' next to server 2 in the table (-192.168.1.1).

Finally, a little bit later again, server 2 becomes the reference server and the DUT is synced to it.

```
awplus#show ntp status
Reference ID   : COA80101 (192.168.1.1) <=== Now referencing server 2
Stratum       : 3
Ref time (UTC) : Mon Nov 26 23:51:01 2018
System time   : 0.000003280 seconds fast of NTP time
Last offset   : +0.000004620 seconds
RMS offset    : 0.000009592 seconds
Frequency     : 17.123 ppm slow
Residual freq : +0.003 ppm
Skew          : 0.339 ppm
Precision     : -20 (0.000000954 seconds)
Root delay    : 0.020674653 seconds
Root dispersion : 0.000725461 seconds
Update interval : 128.9 seconds
Leap status   : Normal
```

### Example 2 - multiple eligible servers

If more than one NTP server is eligible and reachable, the associations table will show the server the device is currently synced with, as well as other candidate servers it could sync with if communication with the current server is lost.

Specifically it can indicate that it has communicated with another server and determined that the server is an acceptable server to sync against. However it won't sync with the other server unless communication with the server currently synced with is lost.

```
awplus#show ntp associations
remote      refid          st t when poll reach  delay  offset  disp
-----
+192.168.2.1 202.46.185.18 2 u  82  256  377  0.173  -0.001  0.001 <== available
*192.168.1.1 202.46.185.18 2 u   6  128  377  0.154  -0.004  0.001 <== synced to
* system peer, # backup, + candidate, - outlier, x false ticker
```

## Understanding the 'show ntp status' command output

The output from the command **show ntp status** will have a Reference ID of '00000000 ()' if the device has **not** synchronized to a server, as shown in the example output below:

```
awplus#show ntp status
Reference ID      : 00000000 ()
Stratum          : 0
Ref time (UTC)   : Thu Jan 01 00:00:00 1970
System time      : 0.000000002 seconds slow of NTP time
Last offset      : +0.000000000 seconds
RMS offset       : 0.000000000 seconds
Frequency        : 16.793 ppm slow
Residual freq    : +0.000 ppm
Skew             : 0.000 ppm
Precision        : 0 (1.000000000 seconds)
Root delay       : 1.000000000 seconds
Root dispersion  : 1.000000000 seconds
Update interval  : 0.0 seconds
Leap status      : Not synchronised
```

- Once it has synchronized, the Reference ID field displays the server it **last** synchronized with. Note, it doesn't indicate that the device is still synchronized with the server:

```
Reference ID      : C0A80101 (192.168.1.1)<=the device last synced with this server
Stratum          : 3
Ref time (UTC)   : Tue Nov 13 00:44:25 2018
System time      : 0.000000003 seconds fast of NTP time
Last offset      : -0.000002949 seconds
RMS offset       : 0.000002949 seconds
Frequency        : 16.793 ppm slow
Residual freq    : -0.513 ppm
Skew             : 0.286 ppm
Precision        : -20 (0.000000954 seconds)
Root delay       : 0.032584600 seconds
Root dispersion  : 0.001336928 seconds
Update interval  : 2.0 seconds
Leap status      : Normal
```

## Losing communication with the clock server

What happens if communication to the clock sever is lost for a short time? Will my device become out of sync?

The answer is generally no - a device will stay very close to in sync for quite a while if communication to the server is lost. Here's a brief explanation of how this happens.

Every time the local time daemon receives a time stamp from the server, it compares it with the current local time. With relatively few timestamps received, the local time daemon calculates a very close approximation of how far/fast the local clock drifts from the server reference clock during the time that elapses between timestamps received. The more timestamps received, the more accurate the drift calculation is.

Using this drift calculation, the time daemon continually adjusts the local clock (far more frequently than the server poll rate). This is how it keeps the time accurate between timestamps received from the server, even if it has lost communication with the server and hasn't received a new timestamp in a while. In simple terms, the time daemon manages the local clock by effectively speeding up or slowing down the clock so it is "ticking" at the same speed as the reference clock.

**Note:** The drift information is stored in Flash memory and will survive a reboot. However, if the device restarts and cannot communicate with the server on startup then the device time will almost certainly be wrong even if it is just by a second or two. The pre-learned/calculated drift information about the local clock (from before the restart) will still be used, and this will help to stop the clock drifting even further out sync. It will also stop the clock drifting closer to the reference time as well. The device time and time management will remain in this state until communication with a server is established again.

## How to configure NTP

The following section provides information about how to configure NTP in the Device GUI and CLI. There are two components to NTP, a server and a client.

- You can either configure your own NTP server, or grab from an NTP pool.
- NTP clients will then receive time from the NTP server.

If time sync is critical to your network, it is highly suggested that you configure your own NTP server with its' own time source.

**Note:** You will need to correct the time on this device periodically to prevent issues with SSL certifications and other time sensitive features.

You may choose to configure a pool, such as pool.ntp.org, to serve as a more general time source.

The following section describes how to configure an NTP server on your own device with its own time source, and how to then set up NTP clients.

## Configuring NTP in the Device GUI

From version 2.16.0 of the Device GUI onwards, you can configure NTP Relationships and Restrictions for NTP client devices using the GUI. This section contains information about how to configure NTP client devices.

### NTP Relationships for Clients

NTP relationships allow you to specify another device by its IP address or hostname, and indicate how this device should interact with it.

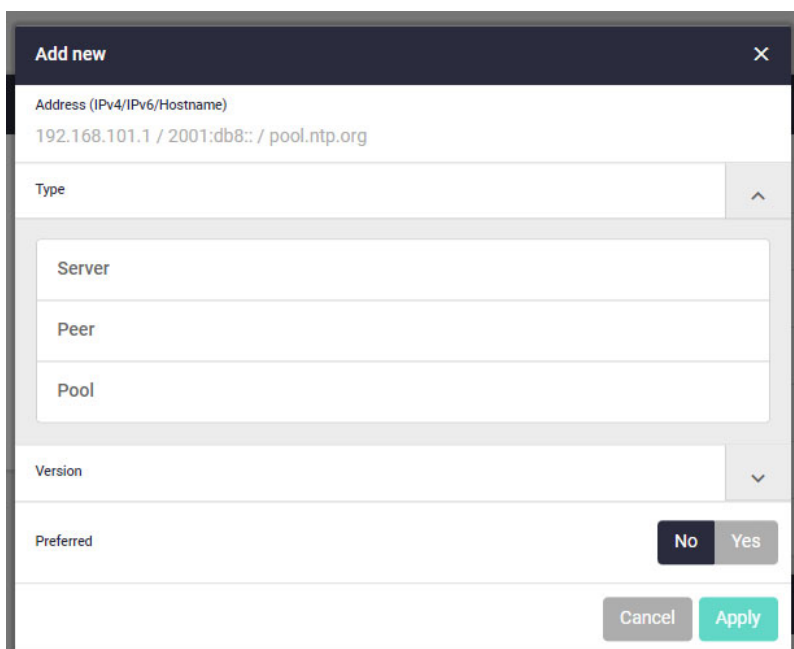
There are three types of relationship:

- **Server** - the other device serves time to this device.
- **Peer** - the other device acts as a peer of this device. If both devices are NTP servers, they will negotiate how to synchronize with each-other.
- **Pool** - the other device is a pool of servers.

We recommend that you use an NTP pool as a time source, such as pool.ntp.org. This is more reliable than a single device.

To add an NTP relationship, click the **+ Add New** button in the NTP Relationships table.

You can then select the NTP version, and select Yes or No for Preferred. When the preferred server is available, NTP uses the preferred server instead of other servers at the same stratum level.



The screenshot shows a dialog box titled "Add new" with a close button (X) in the top right corner. The dialog contains the following fields and options:

- Address (IPv4/IPv6/Hostname):** A text input field containing "192.168.101.1 / 2001:db8:: / pool.ntp.org".
- Type:** A dropdown menu with an upward arrow, showing a list of options: "Server", "Peer", and "Pool".
- Version:** A dropdown menu with a downward arrow.
- Preferred:** A toggle switch with "No" (selected) and "Yes" options.
- Buttons:** "Cancel" and "Apply" buttons at the bottom right.



## NTP Restrictions

NTP restrictions enable you to deny or allow the ability for NTP to send queries or serve time to the target IP. You may want to restrict specific IPs from being able to query or serve network time information in order to secure your network from a malicious IP.

To add an NTP restriction, click the **+ Add New** button next to the title.

Specify whether you would like to restrict the query or serve settings by clicking the toggle button next to the respective action.

**New NTP Restriction**
✕

---

Target IP version

IPv6  IPv4

---

IP Address

Specific address or with /subnet to restrict, leave blank for all IPv4 or IPv6 packets

---

Query

Deny  Allow

---

Serve

Deny  Allow

---

Cancel

Apply

## Configuring NTP in the CLI

This section provides an example that illustrates how to configure two switches, one at a Head Office and one at a Regional Office, to provide a network time service. The Head Office switch is connected to a primary time server and provides the most accurate time information. The switch at the Regional Office uses the Head Office switch as its server to avoid the cost of an additional WAN connection but provides slightly less accurate time information.

- To configure NTP on the switch, an NTP server must be defined. NTP transfers time information in UTC format.
- To set the switch to automatically change the time when summer time starts and ends, enable a summer time offset setting.

Example configuration parameters for a network time service:

Site	Regional Office	Head Office
Switch name	RG1	HO1
IP Address of Switch	10.5.35.114	10.12.25.4
IP Address of Peer	10.5.35.113	172.16.7.3

The NTP feature is **enabled** by default on all switches. Each switch must have a server defined where the switch synchronizes its own internal clock.

**Step 1. Define the NTP server.**

On the Head Office switch, specify a primary NTP time server, use the commands:

```
awplus#configure terminal
awplus(config)#ntp server 172.16.7.3
```

Note that you can also specify an IPv6 address for an NTP server:

```
awplus#configure terminal
awplus(config)#ntp server 2001:0db8:010d::1
```

**Step 2. To specify a VRF when configuring an NTP server.**

If the NTP time server resides within a named VRF you can be configure it with the VRF parameter. The example below shows the VRF named 'blue'. Use the commands:

```
awplus#configure terminal
awplus(config)#ntp server 192.16.7.3 vrf blue
```

**Step 3. Configure the NTP parameters.**

On each switch, the offset of local time from UTC time must be specified. In this example, both switches are in the same time zone (NZST), which is 12 hours ahead of UTC time. Use the following commands on both switches:

```
awplus(config)#clock timezone nzst plus 12
```

**Step 4. Check the NTP configuration.**

Check the NTP configuration on each switch by using the command:

```
awplus#show ntp status
```

## NTP Authentication

The purpose of NTP authentication is to enable the client to authenticate the server, and not vice versa. NTP authentication specifically deals with malicious users attempting to spoof a valid NTP server.

Currently, authentication is performed by configuring an MD5 or SHA-1 key. The server and the client must be both configured to perform authentication and to use the same MD5 or SHA-1 key.

### Configuring authentication on the NTP client

On the client, there are two commands required to configure authentication:

Note: NTP authentication is enabled by default and cannot be disabled.

**Step 1. Create one or more MD5 or SHA-1 keys that can be used for NTP authentication, and designate them as trusted.**

```
awplus(config)#ntp authentication-key <keynumber> {md5|sha1} <key>
[trusted]
```

where the **<keynumber>** is an ID number that will be used in other commands to refer to this key.

**md5|sha1** defines an MD5 or SHA1 key.

The **<key>** is the authentication key. For SHA1, this is a 20 hexadecimal character string.

For MD5, this is a string of up to 31 ASCII characters.

**trusted** allows you to add this key to the list of authentication keys that this server trusts.

**Step 2. When defining the NTP server that the switch wishes to receive time updates from, specify the key that will be used to authenticate the session to this server.**

```
awplus(config)#ntp server <serveraddress> key <keynumber>
```

where the **<keynumber>** is the ID number of an MD5 or SHA-1 key that has been created for NTP authentication, and is in the list of trusted keys.

The server must also be configured to use this key.

## Configuring authentication on the NTP server

When configuring the switch as an NTP server, the configuration required to enable authentication on the server is:

**Step 1. Create an MD5 or SHA-1 key that can be used for NTP authentication, and designate them as trusted.**

```
awplus(config)#ntp authentication-key <keynumber> {md5|sha1} <key>
[trusted]
```

where the **<keynumber>** is an ID number that will be used in other commands to refer to this key.

**md5|sha1** defines an MD5 or SHA1 key.

The **<key>** is the authentication key. For SHA1, this is a 20 hexadecimal character string. For MD5, this is a string of up to 31 ASCII characters.

**trusted** allows you to add this key to the list of authentication keys that this server trusts.

All clients that wish to carry out authenticated sessions with this server must specify this key as the key they will use for sessions to this server.

**Note:** Authentication is initiated by the client. If the client is not configured for authentication then the server will still accept the client's session and serve time updates to the client. But if the client is configured for authentication and the server is not, then the client will refuse to establish the session and will ignore any timestamps sent from that server.

## Authentication and filtering for NTP peers

It is also possible for a pair of NTP peers to authenticate each other. In this case, the configuration is the same as in the NTP client case, except that the final command is replaced by one that defines a peer relationship

```
awplus(config)#ntp peer <peeraddress> key <keynumber>
```

Of course, both peers must use the same key in the session with each other.

### Filtering and authentication can be used together to provide a secure configuration

Filtering limits the addresses from which NTP messages will be accepted, and authentication enables a client to determine that a server is who it says it is.

- Use the **ntp restrict** command with the optional argument 'serve' to allow/deny client or peer requests from specific hosts, or even entire networks. The restriction can specify which addresses it will serve, or refuse to serve time to.

```
awplus(config)#ntp restrict {default-v4|default-v6|<host-address>|<host-subnet>} serve {allow|deny}
```

- The **ntp restrict** command can also specify which addresses it will accept connections from for querying of statistics. This is by using the optional 'query' argument.'

```
awplus(config)#ntp restrict {default-v4|default-v6|<host-address>|<host-subnet>} query {allow|deny}
```

- The **ntp restrict** command can be used without either of the 'serve' or 'query' arguments to allow or deny both to specified addresses.

```
awplus(config)#ntp restrict {default-v4|default-v6|<host-address>|<host-subnet>} {allow|deny}
```

- The way to restrict which servers the switch will accept messages from is via the **ntp server** command. Essentially, the switch will only accept server responses from server addresses added by the **ntp server** command.

```
awplus(config)#ntp server {<serveraddress>|<servername>}
```

# Troubleshooting

**Problem** The switch is not assigning the time to devices on the LAN.

**Solution:**

- Check that the NTP peer's IP address is entered correctly. Check that the NTP peer can reach the switch, by pinging the switch from the NTP peer.

**Problem** The switch's clock does not synchronize with the NTP peer.

**Solutions:**

- The switch's clock can synchronize with the NTP peer only when its initial time is similar to the NTP peer's time (after setting the UTC offset). Manually set the switch's time so that it is approximately correct, and enable NTP again.
- Use the **show ntp counters** command to see if any NTP packets have been dropped.

**Problem** The switch's time is incorrect, even though it assigns the correct time to devices on the LAN.

**Solution:**

- The UTC offset is probably incorrect, or needs to be adjusted for the beginning or end of summer time.