# Contents

## What You Will Learn

```latex
What You Will Learn
```

## Introduction

The Challenge

The Solution

## PTP Concepts

PTP Clock Types

PTP Topology

PTP Algorithm

Hardware and Software Time Stamping

## End-to-End Data Center PTP Deployment with Cisco Nexus 3548

### Topology

End Data Center PTP Deployment with Cisco Nexus 3548

Summary of PTP Options on the Server

Networking Configuration and Best Practices with Cisco Nexus 3548

Cisco Nexus 3548 PTP Configuration

Cisco Nexus 3548 PTP Configuration Validation

## End-to-End Data Center PTP Performance with Cisco Nexus 3548 and Cisco UCS Server

PTP Performance Measurement Definitions and Concepts

Performance Measurement Topology and Equipment

Baseline: Spirent with 48 Ports

Baseline with Higher PTP Rate

Full-Mesh Background Traffic

Congestion and Buffering

Long-Distance Link

Three-Layer Topology

Drift

Software PTP: 44 Servers

Hardware PTP: 44 Servers

## Latency Monitoring Using Cisco Nexus 3548 ERSPAN and PTP Time Stamping

Concept of ERSPAN

ERSPAN with Time-Stamping Configuration

Latency Monitoring Using CorvilNet

Cisco Nexus 3548 ERSPAN Time-Stamp Performance

## About Transparent Clocks and Boundary Clocks

Conclusion

For More Information

About the Author
What You Will Learn

This document describes how to enable a highly accurate timing solution that can provide sub-microsecond accuracy for today’s data center network and financial trading applications by using IEEE 1588-2008 Precision Time Protocol (PTP) Version 2 (PTPv2). PTP is a distributed nanosecond accuracy timing synchronization protocol for packet network.

This document explains the challenges that today’s network and applications are facing and why PTP is needed to provide sub-microsecond accuracy and how the protocol works. It explains PTP concepts and compares hardware and software time stamping. It also presents the PTP features that Cisco Nexus® 3548 Switches support, including PTP time stamping in Encapsulated Remote SPAN (ERSPAN). It provides PTP best practices using the Cisco Nexus 3548 and Cisco Unified Computing System™ (Cisco UCS®) servers for an end-to-end precision data center timing solution. The document also describes the PTP accuracy that the Cisco Nexus 3548 and Cisco UCS can achieve and explains how the measurements are performed.

Introduction

Accurate and precise timing information is critical to today’s data center networks and financial trading applications. Network and system administrators need the visibility to see exactly what is happening on the network and when each event occurs. Application developers and administrators need to correlate various event logs with processes and applications in a very large and complex computing environment. Compliance and digital forensics also require that every data transaction be precisely time stamped. A fundamental requirement for today’s data network is a reliable, accurate, and deployable time-synchronization protocol so that accurate timing information can be provided to all the relevant elements of the data communications network, including routers, switches, servers, and applications.

Today, data center networks are built with switches that are faster than ever. The servers are built with faster CPUs, network interface cards (NICs), memories, and hard drives. Applications on the servers are optimized to run faster and faster. This technology results in significant improvement in overall end-to-end system latency. For example, modern financial exchanges with the latest technology claim that their round-trip market-order latency is less than 100 microseconds. The market-leading Cisco Nexus 3548 ultra-low-latency cut-through switch can forward packets in less than 220 nanoseconds even for 9216 bytes jumbo packets (http://www.cisco.com/en/US/prod/collateral/switches/ps9441/ps11541/ps12581/white_paper_c11-716751.pdf). Now the question is this: can networking timing protocols keep up?

The Challenge

Traditionally, Network Time Protocol (NTP) has been used to provide millisecond-level timing in packet-based networks. However, millisecond accuracy is no longer adequate because of the reasons mentioned in the previous section.

Clearly, to gain visibility into exactly what happens in each process and in the server and switch, organizations need a timing synchronization protocol that can provide microsecond-level details across the whole network.
The Global Positioning System (GPS) can provide accuracy of +/-100 nanosecond (ns), but it needs a dedicated medium to distribute the signal to the end user, which means that every device in the network needs either BNC with IRIG-B or other serial interface to receive the GPS timing information from a separate network. This requirement makes GPS impossible to deploy in a data center network, in which even the smallest server farm has hundreds or thousands of servers, as well as routers, switches, and other network elements that don’t have dedicated special timing protocol interfaces. Practically, organizations need a packet-based solution that is precise and easy to implement and manage.

The Solution
IEEE 1588-2008 PTPv2 is a timing solution for today’s data center and financial applications. It provides the following benefits:

- Spatially localized systems with options for larger systems
- Packet-based timing distribution and synchronization
- Nanosecond to sub-microsecond accuracy
- Low administrative operation, easy to manage and maintain
- Provisions for the management of redundant and fault-tolerant systems
- Low cost, low resource use, works well for both high-end and low-end devices

PTP’s accuracy comes from the hardware support for PTP in the switch and server network interface cards (NICs). It allows the protocol to accurately compensate for message delays and variation across the network. This hardware support for PTP enables the protocol to achieve nanosecond accuracy.

PTP Concepts
This section presents some of the main concepts of PTP.

PTP Clock Types
The main PTP clock types are summarized here.

Grandmaster clock (GM)
A grandmaster clock is the highest-ranking clock within its PTP domain and is the primary reference source for all other PTP elements.

Secondary clock
A secondary clock receives the time information from a primary clock by synchronizing itself with the primary clock. It does not redistribute the time to another clock. In the data center, servers are typically PTP secondary clocks.

Ordinary clock
An ordinary clock is a PTP clock with a single PTP port. It could be a primary clock (grandmaster) or a secondary clock.

Boundary clock (BC)
A boundary clock is the intermediary device between a PTP grandmaster and its PTP secondary clients. It has multiple PTP ports in a domain and maintains the time scale used in the domain. Different ports on the boundary clock can be primary ports or secondary ports. The boundary clock terminates the PTP flow, recovers the clock and time stamp, and regenerates the PTP flow. Effectively, there is a secondary port to recover the clock and primary ports to regenerate the PTP packets.
**Transparent clock (TC)**

A transparent clock measures the time needed for a PTP event message to transit the device and then compensates for the packet delay.

**PTP Topology**

The Best Primary Clock Algorithm (BMCA) is used to select the primary clock on each link, and it ultimately selects the grandmaster clock for the whole PTP domain. It runs locally on each port of the ordinary and boundary clocks to compare the local data sets with the received data from Announce messages to select the best clock on the link. Figure 1 shows an example of a PTP topology in a data center.

![Figure 1. PTP Topology in a Data Center](image)

**Figure 1. PTP Topology in a Data Center**

In an IEEE 1588-2008 PTPv2 network, the primary-secondary hierarchical clock topology needs to be established in a PTP domain before clock synchronization occurs. This tree-like topology is similar to the spanning-tree topology, and the grandmaster clock is the most accurate clock in this hierarchical system, so every PTP secondary clock synchronizes with it. In the PTP network, PTP-enabled ports of the ordinary and boundary clocks examine the contents of all PTP Announce messages received on the port, and then each port runs an independent PTP-state machine to determine the port status. Using BMCA, Announce messages, and the data sets associated with the ordinary or boundary clock, the PTP port can be determined to be in one of the following three states:

- **Primary**: The port is the source of time on the path served by the port. A clock having a primary port becomes the primary clock (not the grandmaster) for its downstream PTP nodes.
- **Secondary**: The port synchronizes with the device on the path on the port that is in the primary state.
- **Passive**: The port is not the primary on the path, nor does it synchronize with a primary. This state prevents timing loops at the PTP level.

Figure 1 shows that when a network has multiple primary clocks - for example, because a new grandmaster clock is added to the system - eventually only one is selected as the grandmaster clock, and it becomes the root of the primary-secondary topology. The port on boundary clock 1 connected to primary clock 2 will transit to a passive state, and no primary-secondary relationship will be established between those two clocks. The dashed line in the figure indicates that the primary-secondary relationship between two boundary clocks is not formed, and that one of the ports is in a passive state as determined by the port state machine.
PTP Algorithm
At a high-level, PTP exchanges packets that contain a time stamp that represents the current time to which the
clock of the receiving device needs to be adjusted. For example, assume that the PTP source sends a PTP
message advertising a time of 1:00:00 p.m. The problem is that this message will take time to reach its destination.
If the PTP packet took 1 second to reach its source, it would be 1:00:01 p.m. when the source receives a PTP
packet indicating 1:00:00 p.m. How does PTP compensate for the network latency?

The synchronization is achieved through a series of messages exchanged between primary and secondary clocks,
as shown in Figure 2.

1. The primary clock sends the Sync message. The time that the Sync message leaves the primary is time
   stamped as t1, which can be embedded in the Sync message itself (one-step operation) or sent in the
   Follow_Up message (two-step operation).
2. The secondary receives the Sync message; t2 is the time that the secondary receives the Sync message.
3. The secondary sends the Delay_Req message, which is time stamped as t3 when it leaves the secondary and
time stamped as t4 when the primary receives it.
4. The primary responds with a Delay_Resp message that contains time-stamp t4.

Figure 2. Clock Synchronization Process
In the following example (Figure 3), the primary time is initially 100 seconds and the secondary time is 80 seconds. The secondary time needs to be adjusted to the primary time.

Figure 3. PTP Message Exchange Example

The clock offset (the difference between the primary and secondary clocks) can be calculated as follows:

\[ \text{Offset} = t_2 - t_1 - \text{meanPathDelay} \]

The link delay from the primary to the secondary is \( t_2 - t_1 \).

The link delay from the secondary to the primary is \( t_4 - t_3 \).

IEEE 1588-2008 assumes that the path delay between the primary and secondary clocks is symmetrical, so the mean path delay is calculated as follows:

\[ \text{meanPathDelay} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \]

You can use this formula to calculate the offset between the primary and secondary in the preceding example:

\[ \text{Mean Path Delay} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} = \frac{(-18 + 22)}{2} = 2 \]

\[ \text{Offset} = t_2 - t_1 - \text{Mean Path Delay} = 82 - 100 - 2 = -20 \]

So in this case, the secondary is 20 seconds behind the primary and will adjust its clock to +20 seconds.

A PTP device that implements this algorithm is known as a two-step clock because the time information is provided by Sync and Follow_Up messages.

Another version of the algorithm, the one-step operation, sends the time stamp in the Sync message instead of using a separate Follow_Up message. The one-step PTP can be more precise, but can be more complicated to implement in hardware. This complexity mainly derives from the need to update the time stamp in real time and to modify the checksum.

The advantage of one-step PTP is that it prevents increased load on both the link and the CPU.
**Hardware and Software Time Stamping**

The objective of PTP implementation on a switch is to provide PTP timing signals to the connected servers so that the system clocks can be synchronized accurately.

In a software PTP implementation, the time stamping of packets is performed by the CPU. This solution is easier to design and implement and is also cost effective compared to hardware time stamping. But it is less precise: packets need to go to the CPU to be time stamped, and the CPU is running the whole operating system, which includes many processes in addition to the PTP process. Even if the PTP process is given the highest priority, software PTP cannot be as precise as hardware PTP.

In a hardware PTP implementation, the capture and insertion of time stamps in PTP packets is performed by the application-specific integrated circuit (ASIC) for a switch or by the NIC for a server. This process ideally is performed at the physical (PHY) level. For ingress, this is right after packets have been received on the wire. For egress, this is right before packets are serialized on the wire, and after packets leave the buffer, so the time stamp precision is not affected by potential buffer use due to congestion or speed mismatch.

**End-to-End Data Center PTP Deployment with Cisco Nexus 3548 Topology**

A typical Data Center PTP topology is as follows. Because PTP is based on Ethernet, it can run over the existing data center architecture and doesn’t usually require a redesign of the data center.

Figure 4 shows a typical leaf-and-spine data center design, with PTP components.

**Figure 4.  Data Center PTP Deployment**

Now let's go over the different components of this end-to-end Data Center PTP deployment.
Grandmaster Clock

The grandmaster clock typically is dedicated, specific equipment. This device usually is connected to GPS, which provides an accurate time source input. The primary clock then generates PTP packets over the network.

The Symmetricom TimeProvider 5000 is an example of a PTP grandmaster clock and is shown in Figure 5.

Figure 5. Symmetricom TimeProvider 5000 PTP Grandmaster Clock

PTP-Enabled Cisco Nexus 3548 Data Center Switch

The Cisco Nexus 3548 Switch is a high-performance, high-density, ultra-low-latency Ethernet switch. This compact one-rack-unit (1RU) 1, 10, and 40 Gigabit Ethernet switch provides **line-rate Layer 2 and 3 switching**. It runs the industry-leading Cisco® NX-OS Software operating system, providing customers with features and capabilities that are widely deployed throughout the world. The Cisco Nexus 3548 is optimized for financial co-location deployments. It provides a robust unicast and multicast routing protocol feature set and has a **190 nanosecond port-to-port latency on all ports** with any packet size and any feature enabled. Figure 6 shows the Cisco Nexus 3548.

Figure 6. Cisco Nexus 3548 Switch

IEEE 1588-2008 PTPv2 is supported on the Cisco Nexus 3548 Ethernet switches starting from Cisco NX-OS Software Release 6.0(2)A1(1). It includes the following features:

- Two-step boundary clock
- User Datagram Protocol (UDP) over IPv4 multicast using multicast address 224.0.1.129 as defined in the IEEE1588 standard
- Hardware-assisted PTP implementation
- Effective handling of network congestion by processing and forwarding PTP messages with higher priority by default; no need for another step to configure additional quality of service (QoS)
- Support for ERSPAN with ERSPAN Header Version 3, which includes a time stamp
- Support for PTP on both Layer 2 and Layer 3 interfaces

The Cisco Nexus 3548 supports PTP operations with hardware assistance. The forwarding ASIC time stamps the PTP packets in both the ingress and the egress direction in hardware.
Figure 7 shows the Cisco Nexus 3548 PTP architecture.

With PTP, the IEEE 1588 packet is time stamped at the ASIC ingress point to record the event message arrival time in hardware at the parser level (1). The time stamp points to the first bit of the packet (following the start-frame delimiter [SFD]) (2). Next, the packet is copied to the CPU with the time stamp and destination port number (3). Then the packet traverses the PTP stack (4). The advanced PTP clock algorithm in the Cisco Nexus 3548 keeps track of all the timing and frequency information and makes necessary adjustments to help ensure accurate time. Finally, the packet is sent out at the egress port and internally marked as a high-priority packet to help ensure priority egress out the switch (5). The corresponding time stamp for the transmitted packet is available from the First In, First Out (FIFO) transmission time stamp.

The Cisco Nexus 3548 PTP implementation is not affected by congestion or buffering. The switch adds the time stamps at the PHY level, when the packet is ready to be serialized and after it has been buffered. This approach results in very high PTP precision, as detailed in the section End-to-End Data Center PTP Deployment with Cisco Nexus 3548.
PTP on the Server

PTP on the server can be either a software implementation or a hardware implementation. Both have advantages.

Software PTP on the Server

With software PTP on the server, the NIC does not have any specific capabilities for PTP hardware time stamping. The PTP time stamping is performed purely in software. Performance is typically within 10 to 100 microseconds, which is already a significant improvement over other timing protocols such as NTP. The section End-to-End Data Center PTP Deployment with Cisco Nexus 3548 later in this document discusses software PTP performance.

The most common Software PTP client for Linux is PTPd, which is open source. PTPd can run on most 32-bit or 64-bit processors and is supported on Linux, uClinux, FreeBSD, and NetBSD. It is a lightweight user-space process. PTPd can achieve microsecond-level accuracy in purely software mode. PTPd and its documentation can be accessed at [http://ptpd.sourceforge.net](http://ptpd.sourceforge.net).

An important benefit of software PTP is that it entails no additional cost because software PTP does not require a specialized PTP-capable NIC.

The drawback of software PTP is its limited accuracy, caused by the operating system latency. If better PTP accuracy is required, hardware PTP solutions should be considered.

Hardware PTP on the Server

Hardware PTP on the Server means that it is equipped with a PTP capable NIC. This NIC allows PTP to achieve much better accuracy.

The ingress and egress time stamping of PTP packets is performed by the clock of the PTP-capable NIC. The clock of the server then synchronizes with the PTP hardware clock on the NIC.

Unlike in a pure software implementation, the PTP packets don't have to reach the CPU to be time stamped. The time stamping is performed at the hardware level when the packets are received and sent. This approach is the reason that hardware PTP on the server can achieve sub-microsecond accuracy.

The PTP software client needs to support hardware PTP. The most common client for hardware PTP is linuxptp, also known as ptp4l.

Ptp4l supports the Linux PTP Hardware Clock (PHC) subsystem by using the clock_gettime() family of calls, including the new clock_adjtimex() system call. It supports hardware and software time stamping through the Linux SO_TIMESTAMPING socket option. With PTP hardware support on the NICs, sub-microsecond accuracy becomes achievable. Ptp4l and its documentation can be found at [http://linuxptp.sourceforge.net](http://linuxptp.sourceforge.net).

On Linux systems, ethtool can be used to determine whether a NIC supports PTP hardware time stamping. If the output includes SOF_TIMESTAMPING_SOFTWARE, SOF_TIMESTAMPING_TX_SOFTWARE, SOF_TIMESTAMPING_RXSOFTWARE, then the NIC supports PTP hardware time stamping.

Summary of PTP Options on the Server

Software PTP is not expensive, but it is not as precise as hardware PTP. Hardware PTP is more precise, but it requires a hardware PTP-capable NIC.

The important point to know is the application accuracy requirement. If it is a multiple of (or some) microseconds, software PTP can be sufficient. If better accuracy (few microseconds) is required, then hardware PTP should be considered.
Networking Configuration and Best Practices with Cisco Nexus 3548

Cisco Nexus 3548 PTP Configuration


Mandatory Configuration

The commands shown here need to be present for PTP to operate on the Cisco Nexus 3548.

Global Configuration Commands

```
switch(config)# feature ptp
```

This command enables PTP on the switch.

```
switch(config)# ptp source <ip>
```

This command specifies the source IP address for the PTP packets generated by the switch.

Interface Configuration Commands

```
switch(config)# interface Ethernet slot/port
switch(config-if)# ptp
```

This command enables PTP on a port. The Cisco Nexus 3548 is a boundary clock, so it has both primary and secondary ports. There is no configuration difference between a primary port and a secondary port. They are both configured with the "ptp" option and BMCA will determine whether the port is a PTP secondary or primary port.

Optional Configuration

```
switch(config)# clock protocol ptp
```

This command configures the switch to use PTP to update the system calendar. This configuration keeps the clock of the switch synchronized with PTP. Not enabling this command won’t prevent the switch from propagating the PTP clock on its primary ports. However, the time source will be the Nexus local clock.

```
switch(config)# interface ethernet slot/port
switch(config-if)# sync interval value
```

This command configures the PTP Sync message packet rate on an interface. If a value is not specified, the default is 0.

The PTP logSyncInterval value represents the number of PTP Sync message packets sent on the interface per second (Table 1).

<table>
<thead>
<tr>
<th>logSyncInterval Value</th>
<th>Number of PTP Sync Messages per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3</td>
<td>8</td>
</tr>
<tr>
<td>-2</td>
<td>4</td>
</tr>
<tr>
<td>-1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1 every 2 seconds</td>
</tr>
</tbody>
</table>

Table 1. Mapping of the Number of PTP Sync Messages per Second to the logSyncInterval Value
You should leave the default sync interval value set to 0 unless you have a specific need for greater accuracy. The section End-to-End Data Center PTP Deployment with Cisco Nexus 3548 later in this document shows accuracy results with both sync interval 0 and -3 on all ports. The Cisco Nexus 3548 supports a sync interval value of -3 on all ports, which offers the highest rate of PTP Sync messages.

**Cisco Nexus 3548 PTP Configuration Validation**

The commands shown here can be used to validate PTP on the Cisco Nexus 3548.

```
switch# show clock
```

This command can be used to verify that the switch clock is synchronized with the grandmaster clock. You cannot verify precise accuracy with this command-line interface (CLI) command, but you can at least verify that the time of day matches the grandmaster, if `clock protocol ptp` was configured.

```
switch# show ptp clock
```

This command displays the properties of the local clock, including clock identity.

**This is an example of show ptp clock output:**

```
switch# show ptp clock
PTP Device Type: Boundary clock
Clock Identity : 30:f7:0d:ff:fe:9b:e2:41
Clock Domain: 0
Number of PTP ports: 42
Priority1 : 255
Priority2 : 255
Clock Quality:
Class : 248
Accuracy : 254
Offset (log variance) : 65535
Offset From Master : 2
Mean Path Delay : 296
Steps removed : 1
Local clock time:Mon Dec 23 09:51:30 2013
```

```
switch# show ptp parent
```

This command displays the properties of the PTP parent. It is useful for verifying the parent clock’s identity.

**This is an example of show ptp parent output:**

```
switch# show ptp parent
PTP PARENT PROPERTIES

Parent Clock:
Parent Clock Identity: 00:b0:ae:ff:fe:02:8f:fb
Parent Port Number: 1
Observed Parent Offset (log variance): N/A
Observed Parent Clock Phase Change Rate: N/A
```
Grandmaster Clock:
Grandmaster Clock Identity: 00:b0:ae:ff:fe:02:8f:fb

Grandmaster Clock Quality:
Class: 6
Accuracy: 33
Offset (log variance): 25600
Priority1: 0
Priority2: 0

switch# show ptp brief

This command displays the PTP state of all interfaces. A PTP port can be in one of the following three states:

- **Primary**: The port is the source of time on the path served by the port.
- **Secondary**: The port synchronizes with the device on the path on the port that is in the primary state.
- **Disabled**: PTP is not enabled on this port.
- **Passive**: The port is not the primary on the path, nor does it synchronize with a primary.

Because the Cisco Nexus 3548 is a PTP boundary clock and supports only one PTP domain, the switch can have only one secondary port. If the switch has one secondary port already, a second port connected to a second grandmaster will be in the passive state. When the first grandmaster or the first secondary port fails, BMCA will move the previously passive port to a secondary state. With this process, grandmaster redundancy can be achieved.

This is an example of `show ptp brief` output:

```
switch# sh ptp brief

PTP port status
-----------------------
Port         State
-
Eth1/1      Slave
Eth1/2      Master
Eth1/3      Master
Eth1/4      Master
Eth1/5      Master
Eth1/6      Master
...
Switch#
```

```
switch# show ptp corrections

This CLI command displays the last few PTP corrections.
```
Here is an example of `show ptp corrections` output:

```
PTP past corrections
--------------------------------------------------------
Slave Port SUP Time Correction(ns) MeanPath Delay(ns)
--------------------------------------------------------
Eth1/46 Mon Dec 23 09:52:11 2013 48581 -1 293
Eth1/46 Mon Dec 23 09:52:12 2013 49318 3 297
Eth1/46 Mon Dec 23 09:52:13 2013 49193 -8 297
Eth1/46 Mon Dec 23 09:52:14 2013 49208 12 298
Eth1/46 Mon Dec 23 09:52:15 2013 48625 -3 298
Eth1/46 Mon Dec 23 09:52:16 2013 47607 -13 295
Eth1/46 Mon Dec 23 09:52:17 2013 49091 0 295
Eth1/46 Mon Dec 23 09:52:18 2013 47961 2 295
Eth1/46 Mon Dec 23 09:52:19 2013 48005 -1 295
Eth1/46 Mon Dec 23 09:52:20 2013 48350 0 296
Eth1/46 Mon Dec 23 09:52:21 2013 48507 -5 292
Eth1/46 Mon Dec 23 09:52:22 2013 48105 2 292
Eth1/46 Mon Dec 23 09:52:23 2013 48188 12 301
Eth1/46 Mon Dec 23 09:52:24 2013 48021 6 301
Eth1/46 Mon Dec 23 09:52:25 2013 48239 -12 296

End-to-End Data Center PTP Performance with Cisco Nexus 3548 and Cisco UCS Server

PTP Performance Measurement Definitions and Concepts

This section describes essential definitions and concepts used to characterize PTP performance.

The **offset** is the difference between the times on two clocks. In the following test results, it is the measure of how accurately the secondary synchronizes with the primary clock. This measurement indicates the amount of inaccuracy brought by the Cisco Nexus 3548 as a boundary clock. Less is better.

The **mean path delay** is the average time taken by PTP frames to travel between primary and secondary. This measurement does not indicate the performance or accuracy of the switch or servers. A small mean path delay is useful for obtaining baseline results. A large mean path delay with a lot of jitter is representative of a complex data center with buffering and latency spikes, control protocols running, a high rate traffic, etc. This measure can be interesting for a real-world PTP performance test in a processing-intensive environment.

The **standard deviation** indicates the amount of variation or dispersion from the average. A low standard deviation indicates that the data points tend to be very close to the mean (also called the expected value); a high standard deviation indicates that the data points are spread out over a large range of values.

In the rest of the document, the following definitions are used to characterize the PTP performance:

- Average offset
- Offset standard deviation
- Minimum and maximum offset peak
- Mean path delay
These values are measured by external tools using the same stable clock reference.

The average offset is often close to 0 because the positive and negative clock offsets equalize each other when calculating an average.

Minimum and maximum offset peaks are useful; however, those values alone do not indicate how often those peaks were reached. Therefore, the offset standard deviation is important in addition to the average offset and offset peaks. Another way to see the offset standard deviation is the jitter.

Those four data points together with the mean path delay provide a very good view of a device’s PTP performance.

Figure 8 is an example of an offset graph in which the average offset, offset standard deviation, and minimum and maximum offset peaks are shown. The vertical axis is the offset value, and the horizontal axis is the set of PTP clients.

Figure 8. Sample Graph of Offset Values

**Performance Measurement Topology and Equipment**

In terms of topology and equipment, there are a few different ways to measure the PTP performance of a device.

The first one is using a traffic generator. It purely measures the performance of the network device under test. It provides an excellent baseline and is a good way to compare network switches to each other. The concept can be compared to an RFC throughput or latency test. However, it does not fully represent a real-world end-to-end scenario for PTP.

The second one is building a data center topology using a set of servers. It is very important to specify the server CPU model and speed, the amount of server memory, Operating System type, NIC card type, and PTP software or hardware characteristics and configuration on the server. It’s a good indication of a real-world data center deployment and the end-to-end PTP accuracy that can be achieved.
**Baseline: Spirent with 48 Ports**

To establish a baseline of PTP performance on the Cisco Nexus 3548, a traffic generator from Spirent was used for PTP message generation, primary and secondary simulation, and background traffic. A Spirent 9U chassis (SPT-9000) with HyperMetrics CV 8 x 10 Gigabit Ethernet Enhanced Small Form Factor (SFP+) cards is used. The Spirent release is 4.33.0086.

The Spirent 9U is connected to a Symmetricom TimeProvider 5000 grandmaster clock. Spirent receives the clock from a timing interface. The Symmetricom TimeProvider 5000 grandmaster gets its time source from an embedded GPS receiver.

The Spirent 9U has one primary port connected to the Cisco Nexus 3548. The rest of ports on the Spirent 9U are PTP secondary ports getting the clock from the Cisco Nexus 3548 PTP primary ports, which redistribute the clock back from the Spirent 9U. Therefore, Spirent can compute the offset between the clock sent from its primary port and the clock received on its secondary ports. This result indicates the accuracy of the Cisco Nexus 3548. There is no background traffic for this test reference.

The Cisco Nexus 3548 uses Cisco NX-OS Software Release 6.0(2)A1(1d).

The Spirent 9U and Cisco NX-OS releases are the same for all subsequent Spirent tests in this report.

Figure 9 shows the Cisco Nexus 3548 performance topology with Spirent.

**Figure 9.** Cisco Nexus 3548 Performance Topology with Spirent

After 12 hours, the results as averages of all ports are:

- Average offset: -1 ns
- Offset standard deviation: 9.8 ns
- Minimum offset peak: -32 ns
- Maximum offset peak: 30 ns
- Average mean path delay: 115 ns
The average offset is very close to 0. The offset standard deviation of 9.8 ns shows that during the duration of the test, the average offset values were clustered closely around the mean. The lowest and highest offsets ever seen during the 12 hours duration of the test were -32 and 30 ns.

Therefore, the Cisco Nexus 3548 introduces an extremely small offset in the PTP client clock synchronization process.

The next tests show how PTP on the Cisco Nexus 3548 performs in more complex test cases.

**Baseline with Higher PTP Rate**

This test is using the same topology and settings on the Spirent traffic generator. On the Cisco Nexus 3548, the PTP sync interval is set to -3 on all ports, which means that eight PTP Sync packets are sent by the switch every second. There is no background traffic.

After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 6.7 ns
- Minimum offset peak: -32 ns
- Maximum offset peak: 30 ns
- Average mean path delay: 116 ns

Those results show that with a higher PTP rate, the accuracy is improved. The offset standard deviation is down to 6.7 ns. Therefore, the standard deviation is lower when other values are equivalent. All subsequent tests in this report are performed with a sync interval value of -3.

**Full-Mesh Background Traffic**

The test topology and parameters for full-mesh background traffic test are exactly the same as for the previous baseline tests, except that in addition the Spirent 9U is configured to generate unicast background traffic to the Cisco Nexus 3548. The traffic load is 99 percent, the frame size is 64 bytes, and the traffic pattern is full-mesh on all 48 ports.

After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 7.9 ns
- Minimum offset peak: -70 ns
- Maximum offset peak: 54 ns
- Average mean path delay: 117 ns

The only significant increase is for the minimum and maximum offset peaks, which rise from -32 ns to -70 ns and from 30 ns to 54 ns, respectively. The offset peak stays below 100 ns. Therefore, the Cisco Nexus 3548 PTP performs very well under high-load, data-intensive traffic over a long period of time.

**Congestion and Buffering**

The topology for the congestion and buffering test is similar to that for the previous tests, but 99% data traffic is sent from two ports to one port. Ports 1 and 2 send 99% 10-Gbps traffic to port 3, and so on. This behavior creates congestion on the switch.
Output ports that are congested experience packet drops, which is expected because 20-Gbps of traffic is sent to 10-Gbps. Spirent counters are monitored to verify that data drops are occurring to confirm the congestion behavior. Active buffer monitoring is also used on the Cisco Nexus 3548 to confirm that buffers are being fully utilized.

After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 7 ns
- Minimum offset peak: -70 ns
- Maximum offset peak: 69 ns
- Average mean path delay: 116 ns

Again, the average offset value is null. The offset standard deviation of 7 ns shows that the average offset values were clustered closely around the mean. Offset peaks are below 100 ns. Therefore, the Cisco Nexus 3548 shows very strong PTP performance when its links are congested at their maximum even for a very long period of time.

**Long-Distance Link**

Networks can be spread across multiple remote sites. Sometimes there’s only one PTP grandmaster in one site, but a need to deploy PTP in many sites. Those sites are connected with a long-distance link.

This test simulates site-to-site PTP with a 25-km link. The first Cisco Nexus 3548 Switch represents the switch that would be in the first site. This switch has a long-distance connection to a second Cisco Nexus 3548, which would be in the second site. There is no background traffic in this test. **Figure 10** shows the topology for the long-distance link performance test.

**Figure 10.** Cisco Nexus 3548 PTP Long-Distance Performance Topology
After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 7.8 ns
- Minimum offset peak: -73 ns
- Maximum offset peak: 112 ns
- Average mean path delay: 115 ns

These results show that the Cisco Nexus 3548 still maintains very good PTP performance in a two-layer deployment using a long-distance link between the two Cisco Nexus 3548 Switches.

**Three-Layer Topology**

Many data centers have multiple layers between the grandmaster clock and the servers. Therefore, it is important to test PTP performance in a multilayer scenario.

This test uses three Cisco Nexus 3548 Switches connected in a cascaded topology. As in the previous tests, each switch acts a PTP boundary clock. The last switch is connected back to Spirent to measure the overall offset in the three-layer topology. Figure 11 shows the three-layer performance topology.

**Figure 11.** Cisco Nexus 3548 PTP Three-Layer Performance Topology

After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 8.4 ns
- Minimum offset peak: -57 ns
- Maximum offset peak: 55 ns
- Average mean path delay: 116 ns
The average offset value is null, and the standard deviation is less than 10 ns. The offset peaks stay under 60 ns. Because PTP boundary clock devices perform clock recovery, they could technically bring a larger offset value when connected in a cascaded topology. This is not the case for the Cisco Nexus 3548; its offset value stays very low in a multilayer topology.

**Drift**

In a PTP deployment, the PTP grandmaster could fail, or the link could be lost. Even in a dual-grandmaster deployment, network problems could cause a loss of connection to both grandmasters. Therefore, it is important to measure the way that a PTP device behaves when it loses its connection to its PTP reference source.

To measure the drift of the Cisco Nexus 3548 when it loses its PTP source and goes into holdover, the same topology is reused. There is no background traffic. On the Cisco Nexus 3548, the link to the primary port is shut down. This shutdown is confirmed by checking the link status on the Spirent chassis. On the Cisco Nexus 3548, the CLI command “show ptp parent” is also used to verify that the switch is not connected to any PTP primary. Figure 12 shows the drift measurement topology.

**Figure 12.** Cisco Nexus 3548 PTP Drift Performance Topology

After 12 hours, the results as averages of all ports are:

- Average offset: -1.3 ns
- Offset standard deviation: 6.4 ns
- Minimum offset peak: -99 ns
- Maximum offset peak: 57 ns
- Average mean path delay: 116 ns

The average offset is -1.3 ns compared to 0 ns with a clock reference. The standard deviation did not change significantly. Offset peaks stay under 100 ns. These values show that the Cisco Nexus 3548 PTP clock drifts very little in the absence of a clock reference.
Software PTP: 44 Servers

The software PTP test uses 44 servers running software PTP. The goal is to measure the performance of PTP on those servers. As in the previous tests, the Cisco Nexus 3548 is a boundary clock that receives a PTP reference from a grandmaster. The servers are directly connected to the Cisco Nexus 3548. The switch acts as a PTP primary for the 44 servers. Figure 13 shows the test topology.

Figure 13. 44 Servers Running Software PTP Performance Topology

The Cisco Nexus 3548 is running Cisco NX-OS 6.0(2)A1(1c). All 44 10-Gbps ports are configured as Layer 3 ports with unique IPv4 address and subnet for each interface. The PTP feature is configured globally, and each interface is configured with a sync interval of 0. There is no background traffic.

The servers are Cisco UCS C210 M3 Servers, with Mellanox ConnectX-2 NICs and running open source PTPd software as the software PTP clients. Because the NIC doesn’t support PTP in hardware, all the PTP processing occurs in software. The operating system is Red Hat Enterprise Linux (RHEL) 6.3. All servers are identical in every aspect (CPU model, number of cores, memory, etc.).

The software PTP client running on the servers is PTPd2 built from the source code taken on April 29, 2013, from http://sourceforge.net/projects/ptpd2. PTPd2 is run with the default options.

After 12 hours, the results as averages of all ports are:

- Average offset: -0.7 ns
- Offset standard deviation: 6.8 microseconds
- Minimum offset peak: -47 microseconds
- Maximum offset peak: 51 microseconds
- Average mean path delay: 87 ns
The main result to note here is that although the average offset is close to 0 ns, the offset standard deviation and offset peaks are in the microseconds range. This is a big improvement over typical NTP performance (usually in milliseconds). It also shows that although the Cisco Nexus 3548 has nanosecond PTP accuracy, the server is causing a lot of offset with a pure software PTP implementation.

Hardware PTP on the server is required for better PTP accuracy.

**Hardware PTP: 44 Servers**

The hardware PTP test uses 44 servers running hardware PTP. The goal is to measure the performance of PTP on those servers. As in the previous tests, the Cisco Nexus 3548 is a boundary clock receiving a PTP reference from a grandmaster. The servers are directly connected to the Cisco Nexus 3548. The switch acts as a PTP primary for the 44 servers. Figure 14 shows the test topology.

![Figure 14. 44 Servers Running Hardware PTP Performance Topology](image)

The Cisco Nexus 3548 is running Cisco NX-OS 6.0(2)A1(1c). All 44 10-Gbps ports are configured as Layer 3 ports with a unique IPv4 address and subnet for each interface. The PTP feature is configured globally, and each interface is configured with a sync interval of 0. There is no background traffic.

The servers are Cisco UCS C210 M3 Servers. All servers are identical in every aspect (CPU model, numbers of cores, memory, etc.). The servers are equipped with the Intel I350 NIC with hardware PTP time-stamping support. The operating system is RHEL 6.4.

The PTP client is ptp4l (also known as Linux PTP) Version 1.3. Ptp4l is configured as follows:

```
# Default Data Set
#
twoStepFlag 1
slaveOnly 1
priority1 255
```
priority2 255
domainNumber 0
clockClass 248
clockAccuracy 0xFE
offsetScaledLogVariance 0xFFFF
free_running 0
freq_est_interval 1
#
# Port Data Set
#
logAnnounceInterval 1
logSyncInterval 0
logMinDelayReqInterval 0
logMinPdelayReqInterval 0
announceReceiptTimeout 3
#
# Run time options
#
assume_two_step 0
logging_level 6
path_trace_enabled 0
follow_up_info 0
use_syslog 1
verbose 0
#
#
# Servo Options
#
pi_proportional_const 0.0
pi_integral_const 0.0
pi_offset_const 0.0
clock_servo pi
#
# Transport options
#
transportSpecific 0x0
ptp_dst_mac 01:1B:19:00:00:00
p2p_dst_mac 01:80:C2:00:00:0E
#
# Default interface options
#
network_transport UDPv4
delay_mechanism E2E
time_stamping hardware
After 12 hours, the results as averages of all ports are:

- Average offset: 0 ns
- Offset standard deviation: 40.1 ns
- Minimum offset peak: -247 ns
- Maximum offset peak: 378 ns
- Average mean path delay: 732 ns

The average offset value is null, the offset standard deviation is below 50 ns, and the offset peaks are in the range of 250 to 400 ns. Therefore, a hardware PTP solution on the server in combination with the Cisco Nexus 3548 provides sub-microsecond end-to-end data center PTP accuracy.

**Latency Monitoring Using Cisco Nexus 3548 ERSPAN and PTP Time Stamping**

**Concept of ERSPAN**

ERSPAN transports mirrored traffic over an IP network. The traffic is encapsulated at the source router and is transferred across the network. The packet is decapsulated at the destination router and then sent to the destination interface.

ERSPAN consists of an ERSPAN source session, routable ERSPAN generic routing encapsulation (GRE)-encapsulated traffic, and an ERSPAN destination session. You separately configure ERSPAN source sessions and destination sessions on different switches. Figure 15 shows an ERSPAN topology.

**Figure 15.** ERSPAN Topology

The Cisco Nexus 3548 supports ERSPAN Type III, which contains time-stamp information that can be used to calculate packet latency between different parts of the network.

**Figure 16** shows a typical example of latency monitoring in a data center. The goal is to determine the latency between switch A and switch B.
To monitor the latency, an ERSPAN source session is configured on each switch. The source port for this session is the ingress port of the flow for which the latency is monitored. The destination IP address of the session is the IP address of a server running a packet-capture analyzer such as Wireshark.

The two switches have their clocks synchronized using PTP, so the time stamps recorded in the ERSPAN header have the same time reference.

On the analyzer server, the packets from both switches are received (because the same ERSPAN destination IP address was configured). From there it’s easy to compute the delta between the ERSPAN timestamps of the packets received to have the network latency.

Figure 16 shows how to configure ERSPAN and PTP to perform latency monitoring in a typical network.

Figure 16. Latency Monitoring Using ERSPAN and PTP

The important point to note is that an ERSPAN destination session is not configured on any switch in this scenario. An ERSPAN destination decapsulates the GRE and ERSPAN headers from the packet. Therefore, the time stamp would be lost. Instead, Wireshark is used, and it can read the ERSPAN header and display the time stamp.
Figure 17 shows an example of a Wireshark capture of an ERSPAN packet. The time stamp is highlighted in red.

Figure 17. Wireshark ERSPAN Capture

ERSPAN with Time-Stamping Configuration

ERSPAN header type 3 contains a 32-bit time stamp. This time stamp represents the time of day since January 1, 1970. The CLI option required to enable the ERSPAN time stamp is header-type 3 in the ERSPAN session configuration.

A sample ERSPAN configuration with header-type 3 is shown here:

```
switch(config)# monitor session 1 type erspan-source
switch(config-erspan-src)# vrf default
switch(config-erspan-src)# source interface Ethernet1/30 rx
switch(config-erspan-src)# destination ip 192.168.8.1
switch(config-erspan-src)# header-type 3
switch(config-erspan-src)# no shut
```
ERSPAN time-stamp granularity is an optional configuration that determines the level of detail of the time stamp in the ERSPAN packet. It is configured globally. Options are ns, 100_ns, and 100_ms.

```
switch(config)# monitor erspan granularity {100_ms | 100_ns | ns}
```

The granularity option determines the granularity of the time stamp: either 100 ms, 100 ns, or nanosecond precision. It also determines where the time stamp is recorded.

- With the ns granularity CLI option configured, the time stamp is recorded on ingress. The time stamp is recorded when the first byte of the original packet is received.
- With the 100_ns or 100_ms granularity CLI option configured, the time stamp is recorded on egress. The time stamp is recorded when the first byte of the ERSPAN egress packet is sent.

The Cisco Nexus 3548 is precise at the nanosecond level for the time stamp. The ERSPAN time stamping is performed in hardware.

Note that the PTP time stamp and the ERSPAN time stamp mean different things:

- The PTP time stamp is the time stamp used in PTP control packets that allow PTP to operate and synchronize clocks over the network.
- The ERSPAN time stamp is the time stamp in the ERSPAN header. It is separate from the PTP time stamp. ERSPAN time stamping can actually work without PTP enabled on the switch. In such a case, the ERSPAN time stamp would be derived from the local oscillator of the switch and not be synchronized with any source. It is an acceptable solution to perform basic latency monitoring using just one switch because it would still show the delta between the arrival time of packets in the switch (or the departure time of ERSPAN packets with 100_ns or 100_ms granularity configured).

**Latency Monitoring Using CorvilNet**

CorvilNet is an operational performance monitoring system that provides time synchronized data acquisition, high velocity data transformation, and operational performance monitoring for applications and networks. It provides a latency navigator that can display graphs of latency. This achieved using a similar concept as explain in the previous section, except that the ERSPAN destination IP is the one of CorvilNet. CorvilNet is also able to correlate the latency data with the buffer usage from Active Buffer Monitoring.

The different switches in the network are synchronized using PTP, and therefore the ERSPAN time stamp is used to determine end-to-end latency between the switches.


**Cisco Nexus 3548 ERSPAN Time-Stamp Performance**

This section describes the performance of Nexus 3548 ERSPAN timestamping in conjunction with PTP.

In order to measure it, the following topology is put in place. A GPS and Symmetricom are used. Symmetricom is receives the signal from the GPS and acts a PTP Grandmaster for the rest of the network. The Symmetricom TP5000 is connected to a Spirent chassis.

There are three Nexus 3548 in this topology.

From the PTP perspective, N3548-2 and N3548-3 are PTP secondary in the same PTP domain (N3548-2 acts as a boundary clock between Spirent and N3548-3).
From the data perspective, Spirent sends unicast traffic to N3548-1. Warp SPAN is configured on N3548-1 so that this unicast traffic gets duplicated to both N3548-2 and N3548-3 simultaneously (Multicast could have been used, too).

N3548-2 and N3548-3 are both configured with ERSPAN source sessions. The ingress source port for those sessions is the port connected to N3548-1. The ERSPAN destination IP is a UCS server running Wireshark. **Figure 18** shows the ERSPAN time-stamp performance setup.

**Figure 18.** ERSPAN Timestamp Performance Setup

The goal of the test is to measure the difference between the ERSPAN timestamps sent by N3548-2 and N3548-3, using Wireshark on the UCS server.

On Spirent, incrementing payload is configured to confirm the packet order from two different ports on Wireshark. **Figure 19** and **Figure 20** below are screen captures of Wireshark receiving packets from N3548-2 and N3548-3. Interface 0 and interface 1 are the UCS ports connected to N3548-2 and N3548-3 respectively. The uniquely incrementing payloads allow to identify the unique packets coming from N3548-2 and N3548-3.

In this example, the timestamp from the N3548-2 packet is 40507952 nanoseconds and the timestamp from the N3548-3 packet is 40507959 nanoseconds. Therefore, the ERSPAN time-stamp accuracy for these packets is 7 nanoseconds.
Figure 19. Screen Image of Wireshark for the First Interface

Figure 20. Screen Image of Wireshark for the Second Interface
Spirent is configured to send 10,000 packets. A simple Perl script on the server computes the time-stamp delta from Wireshark PCAP files for the same packets arriving from the two Cisco Nexus 3548 Switches.

The results are as follows:

- The average time-stamp delta is 5 nanoseconds.
- The peak value of the time-stamp delta is 10 nanoseconds.

These results show that the Cisco Nexus 3548 PTP implementation is extremely accurate, and that the time-stamping capability in the Cisco Nexus 3548 ERSPAN feature is very accurate as well.

**About Transparent Clocks and Boundary Clocks**

In cascaded topologies, PTP accuracy can be affected by the accumulation of time dispersion noise due to chained servo. The goal of transparent clocks is to reduce this inaccuracy.

As explained earlier, a boundary clock performs clock recovery. A transparent clock operates differently: it computes how much time packets take to traverse the device (residence time of PTP messages), and update it in the correction field of the PTP message or an associated follow-up message.

In comparison, a boundary clock does not calculate how much time the packets take to traverse the switch. It does not need to, because it will regenerate a whole new PTP message exchange on its secondary ports.

The ports of a transparent clock have no state because the transparent clock does not need to synchronize with the grandmaster clock.

Another benefit of transparent clocks is that they require less time to initialize and reconfigure if network topology changes, because transparent clocks do not run BMCA.

A PTP deployment in the data center typically has fewer than five layers between the grandmaster and servers. The Cisco Nexus 3548 performance tests that used multiple layers show PTP accuracy of less than 10 ns. Therefore, the Cisco Nexus 3548 boundary clock implementation is extremely relevant for PTP data center deployments.

**Conclusion**

IEEE1588-2008 PTPv2 provides a reliable, highly accurate distributed time synchronization solution for the data center, which requires nanosecond or sub-microsecond accuracy. PTP is easy to implement with very little administrative effort and can tolerate network and clock failure with built-in fault-tolerant mechanisms.

The Cisco Nexus 3548 has an extremely precise PTP implementation. Under the most stressful test conditions, Cisco Nexus 3548 PTP accuracy stays under 10 nanoseconds.

This can be combined with a Software PTP implementation on the server to achieve microsecond accuracy.

Hardware PTP on the server can also be used if sub-microsecond accuracy is required on the server. Cisco UCS servers equipped with a hardware PTP-capable NIC and used in combination with the Cisco Nexus 3548 provide end-to-end PTP accuracy performance of less than 50 nanoseconds.

PTP in combination with ERSPAN on the Cisco Nexus 3548 provides a very powerful and easy-to-use solution to perform latency monitoring in the data center with sub-10 nanosecond accuracy.
For More Information


For more information about the Cisco Nexus 3548 Switch, see the detailed product information at the product homepage at http://www.cisco.com/go/nexus3548.


About the Author

Nicolas Delecroix is a Technical Marketing Engineer in the Data Center Group Business Unit at Cisco. He currently focuses on IEEE 1588 PTP, Nexus switching architecture, and ultra low latency solutions.