Stream: Internet Engineering Task Force (IETF)
RFC: 10030
Category: Standards Track
Published: August 2026
ISSN: 2070-1721
Author: M. Lichvar
Red Hat

RFC 10030
Network Time Protocol (NTP) over the Precision Time Protocol (PTP)

Abstract

This document specifies a transport for the client-server and symmetric modes of the Network Time Protocol (NTP) that encapsulates NTP messages in messages of the Precision Time Protocol (PTP). This transport enables hardware timestamping in network interface controllers (NICs) that can timestamp only PTP messages and delay corrections in PTP transparent clocks.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc10030.

Table of Contents

1. Introduction

The Precision Time Protocol (PTP) [IEEE1588-2019] was designed for highly accurate synchronization of clocks in local networks. It relies on hardware timestamping support in all network devices involved in the synchronization (e.g., network interface controllers (NICs), switches, and routers) to eliminate the impact of software, processing, and queueing delays on the accuracy of offset and delay measurements.

PTP was originally designed for multicast communication. Later, support for unicast messaging was added, which is useful in larger networks with partial on-path PTP support (e.g., telecom profiles G.8265.1 [G8265-1] and G.8275.2 [G8275-2]).

The Network Time Protocol (NTP) [RFC5905] does not rely on hardware timestamping support, but implementations can use it if it is available to avoid the impact of software, processing, and queueing delays, similarly to PTP. When comparing PTP with the timing modes of NTP, PTP is functionally closest to the NTP broadcast mode.

An issue for NTP is hardware that can specifically timestamp only PTP packets. This limitation comes from a hardware design that can provide receive timestamps only at a limited rate instead of the maximum rate possible at the network link speed. To avoid missing receive timestamps when the interface is receiving other traffic at a high rate, a filter is implemented in the hardware to inspect each received packet and capture a timestamp only for packets that need it.

The hardware filter can be usually configured for specific PTP transports (e.g., UDP over IPv4, UDP over IPv6, and 802.3) and sometimes even the PTP message type (e.g., sync message or delay request) to further reduce the timestamping rate on the server or client side in the case of multicast messaging, but it typically cannot be configured to timestamp NTP messages sent to the UDP port 123.

Another issue for NTP is missing hardware support in network switches and routers. With PTP, the devices operate as either boundary clocks or transparent clocks. Boundary clocks are analogous to NTP clients that work also as servers for other clients. Transparent clocks are much simpler. They only measure the delay in the forwarding of PTP packets and write this delay to the correction field of either the packet itself (one-step mode) or a later packet in the PTP exchange (two-step mode). Transparent clocks are specific to the PTP delay mechanism used in the network, either end to end (E2E) or peer to peer (P2P).

This document specifies a new transport for NTP to enable hardware timestamping on NICs that can timestamp only PTP messages and to take advantage of one-step E2E PTP unicast transparent clocks. It adds a new type-length-value (TLV) for PTP to contain NTP messages and a new extension field for NTP to provide clients and peers with the correction of their NTP requests from transparent clocks. The NTP broadcast mode is not supported.

The use of PTP messages requires that protocol rules of IEEE 1588 [IEEE1588-2019] be followed. NTP over PTP does not require other PTP clocks to be present in the network. It does not disrupt their operation if they are present. If the network uses one-step E2E transparent clocks, NTP clients and peers using PTP for transport can reach the same or better accuracy as PTP clocks using PTP for synchronization. Hosts in a network can use PTP for synchronization in one domain and transport of NTP messages in another domain at the same time.

1.1. Comparison with PTP

The client-server mode of NTP, even with the PTP transport, has multiple advantages over PTP using multicast or unicast messaging:

- NTP is more secure. Existing security mechanisms specified for NTP such as Network Time Security [RFC8915] still work over the PTP transport. It is more difficult to secure PTP against delay attacks because the sync message is not an immediate response to a client request. The PTP unicast mode allows an almost-infinite traffic amplification, which can be exploited for denial-of-service attacks and can only be limited by security mechanisms requiring client authentication.
- NTP is more resilient to failures. Each client can use multiple servers and detect failed sources in its source selection. In PTP, a single hardware or software failure can disrupt the whole PTP domain. Multiple independent domains have to be used to handle any failure.

- NTP is better suited for synchronization in networks that do not have full on-path PTP support or where timestamping errors do not have a symmetric distribution (e.g., due to sensitivity to the network load). NTP does not assume network delay is constant and the rate of measurements in opposite directions is symmetric. It can filter the measurements more effectively and is not sensitive to asymmetrically distributed network delays and timestamping errors. PTP has to measure the offset and delay separately to enable multicast messaging, which is needed to reduce the transmit timestamping rate.
- NTP needs fewer messages to get the same number of timestamps. It uses less network bandwidth than PTP using unicast messaging.
- NTP provides clients with an estimate of the maximum error of the clock (root distance).

The disadvantage of NTP is that the transmit timestamping rate increases as the number of clients grows. A server that is limited by the hardware timestamping rate cannot provide a highly accurate time service to the same number of clients as with PTP using multicast messaging.

1.2. Requirements Language

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. PTP Transport for NTP

A new TLV is defined for PTP to contain NTP messages in the NTP client (3), server (4), and symmetric modes (1 and 2) (see [RFC5905]). Using other NTP modes in the TLV is not specified. Any transport specified for PTP that supports unicast messaging, and an IPv4 or IPv6 mapping, can be used for NTP over PTP.

The NTP TLV **MUST** be included in a unicast PTP event message. An event message is required to enable the PTP-specific hardware timestamping and corrections of transparent clocks. The PTP message **MUST** conform to PTP version 2 [IEEE1588-2008], PTP version 2.1 [IEEE1588-2019], or any future version of the PTP specification that allows the NTP TLV to be included as an organization-specific TLV.

The NTP TLV is an organization-specific TLV having the following fields (with octets in network order):

- type is 0x8000 (ORGANIZATION_EXTENSION_DO_NOT_PROPAGATE) in PTP version 2.1 or 0x0003 (ORGANIZATION_EXTENSION) in PTP version 2
- lengthField is 8 + length of the NTP message
- organizationId is 00-00-5E (the Organizationally Unique Identifier (OUI) is assigned to IANA by the IEEE Registration Authority)
- organizationSubType is 0x1

- dataField contains two zero octets for 32-bit alignment followed by the NTP message, which would normally be the UDP payload

An NTP client or peer using the PTP transport sends NTP requests contained as the NTP TLV in PTP messages.

An NTP server or peer responding to an NTP request received over the PTP transport **MUST** form its response as the NTP TLV using the same PTP transport. To avoid traffic amplification, the server or peer **MUST NOT** send the response if the PTP message containing the NTP response is longer than the PTP message containing the NTP request. This requirement impacts Autokey [RFC5906], where some responses are longer than the requests (e.g., during certificate exchange). The request **SHOULD** be padded with the PTP PAD TLV (type 0x8008) to the maximum expected length of the response to enable the transmission of the response.

If the NTP response is expected to be used for synchronization (e.g., it is not an error message), the PTP message containing the NTP response **SHOULD** have the same length as the PTP message containing the NTP request, using the PTP PAD TLV if needed, to avoid an asymmetric delay in networks without full on-path PTP support.

The PTP version 2.1 [IEEE1588-2019] specification states the following:

> A domain shall define the scope of PTP message communication, state, operations, data sets, and timescale. Within a PTP Network, a domain is identified by two attributes: domainNumber and sdoId.

In the context of NTP over PTP version 2.1, this means that the NTP servers, clients, and peers **MUST** verify that received PTP messages have the domainNumber and sdoId that are expected to be used by NTP over PTP in the network. The domainNumber **SHOULD** be 123 by default, and sdoId **SHOULD** be 0. The domainNumber 123 is not commonly used by PTP profiles, so it is less likely to interfere with any other PTP operation that might be running in the network. The domainNumber **SHOULD** be configurable to allow moving NTP over PTP to another domain if a conflict with a PTP profile using this domainNumber and sdoId needs to be avoided. However, all servers, clients, and peers using NTP over PTP in the network need to use the same domainNumber and sdoId to be able to communicate with each other.

If the UDP transport is used for PTP, the UDP source and destination port numbers **SHOULD** be the PTP event port (319). If the client implemented port randomization [RFC9109], requests and/or responses would not get a hardware receive timestamp due to the hardware filter matching only the PTP event port.

Any authenticator fields included in the NTP messages **MUST** be calculated only over the NTP message following the header of the NTP TLV. Other data in the PTP message (outside of the NTP TLV) are not protected. With the exception of the PTP correction field requiring special handling as described in the following section, the other PTP fields are used only for the transport of the NTP message and have no impact on the security of NTP, similarly to the IP and UDP headers.

Receive and transmit timestamps contained in the NTP messages **SHOULD NOT** be adjusted for the beginning of the NTP data in the PTP message. To minimize the impact of different link speeds on accuracy in networks without full on-path PTP support, the transmit timestamp **SHOULD** correspond to the PTP message timestamp point (i.e., the beginning of the first symbol after the Ethernet start of frame delimiter), and the receive timestamp **SHOULD** be transposed from the PTP message timestamp point to the ending of the reception (e.g., the ending of the last symbol of the Ethernet frame check sequence).

3. Network Correction Extension Field

One-step E2E PTP transparent clocks modify the correction field in the header of the PTP event messages containing NTP messages. To be able to verify and apply the corrections to an NTP measurement, the client or peer needs to know the correction of both the request and response. The correction of the response is in the PTP header of the message itself. The correction of the request is provided by the server or other peer in a new NTP extension field included in the response.

The format of the Network Correction Extension Field is shown in Figure 1.

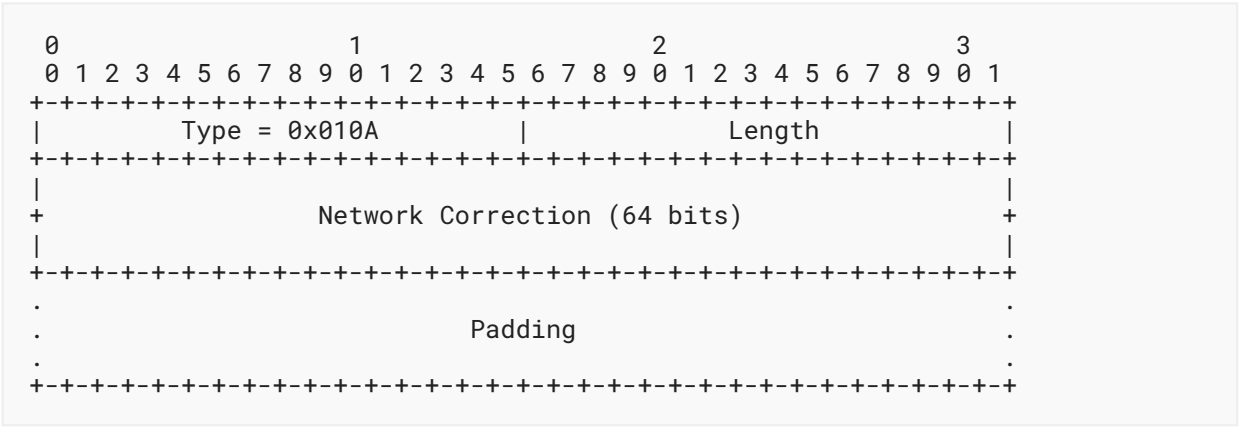

Figure 1: Format of Network Correction Extension Field

The length of the padding is the minimum required to make a valid extension field in the used version of NTP. In NTPv4, it is 16 octets to get a 28-octet extension field conforming to [RFC7822].

The Network Correction field in the extension field uses the 64-bit NTP timestamp format (with resolution of about 1/4th of a nanosecond). The correction field in the PTP header has a different format (64-bit nanoseconds + 16-bit fraction).

The value of the NTP network correction is the sum of PTP corrections provided by transparent clocks and the time it takes to receive the packet (i.e., packet length including the frame check sequence divided by the link speed).

The reason for not using the PTP correction alone is to avoid an asymmetric correction when the server and client, or peers, are connected to the network with different link speeds. The receive duration included in the NTP correction cancels out the transposition from the PTP receive timestamp (which corresponds to the beginning of the reception) to NTP receive timestamp (which corresponds to the end of the reception).

The Figure 2 shows the NTP timestamps, transmit/receive durations, and processing and queuing delays included in PTP corrections for an NTP exchange made over two PTP transparent clocks. The link speed is increasing on the network path from the client to the server. The propagation delays in cables are not shown.

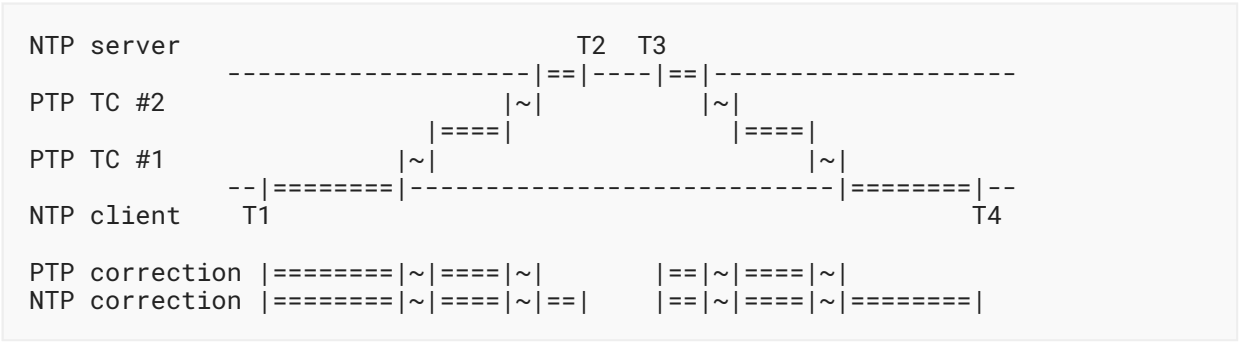

Figure 2: PTP Versus NTP Correction

When an NTP server that supports the PTP transport receives an NTP request containing the Network Correction Extension Field, it **SHOULD** respond with the extension field providing the network correction of the client's request. The server **MUST** ignore the value of the network correction in the request.

An NTP client or peer that supports the PTP transport and is configured to use the network correction for the association **SHOULD** include the extension field in its NTP requests. In the case of a client, the correction value in the extension field **SHOULD** be always zero.

When the client or peer has the network correction of both the request and response, it can correct the measured NTP peer delay and offset:

- delta_c = delta - (nc_rs + nc_rq - dur_rs - dur_rq) * (1 - freq_tc)
- theta_c = theta + (nc_rs - nc_rq) / 2

where

- delta is the NTP peer delay from [RFC5905]
- theta is the NTP offset from [RFC5905]
- nc_rq is the network correction of the request
- nc_rs is the network correction of the response
- dur_rq is the transmit duration of the request
- dur_rs is the receive duration of the response

- freq_tc is the maximum assumed frequency error of transparent clocks

The corrected delay (delta_c) and offset (theta_c) **MUST NOT** be accepted for synchronization if any of delta_c, nc_rs, and nc_rq is negative. This requirement limits the error caused by faulty transparent clocks and on-path attackers.

Root delay (DELTA) **MUST NOT** be corrected to ensure that the maximum assumed error (root distance) remains independent of network corrections.

The scaling by the freq_tc constant (e.g., 100 parts per million (ppm)) is needed to make room for errors in corrections made by transparent clocks running faster than true time and to avoid samples with larger corrections from getting a shorter delay than samples with smaller corrections, which would negatively impact their filtering and weighting.

The dur_rq and dur_rs values make the corrected peer delay correspond to a direct connection to the server. If they were not used, a perfectly corrected delay on a short network path would be too close to zero and frequently negative due to frequency offset between the client and server. Note that NTP peers and PTP clocks using the E2E delay mechanism are more sensitive to frequency offsets due to longer measurement intervals. If dur_rq is unknown, it **MAY** be assumed to be equal to dur_rs.

4. IANA Considerations

4.1. New IANA PTP TLV Subtypes Registry

IANA has created the "IANA PTP TLV Subtypes" registry under the "IANA OUI Ethernet Numbers" registry group for organizationSubType values of PTP TLVs using 00-00-5E as the organizationId (i.e., the OUI assigned to IANA by the IEEE Registration Authority).

The entries in the registry have the following fields, which are **REQUIRED**:

Subtype: An integer in the range 0-0xFFFFFF

Description: A short text description

Reference: A reference to a document describing the IANA PTP TLV

The subtype range is split into the following three ranges with different allocation policies:

0-0xFFFF: IETF Review

0x10000-0x7FFFFF: Specification Required

0x800000-0xFFFFFE: Experimental and Private Use

The initial contents of the registry are as follows:

Subtype	Description	Reference
0x0	Reserved	RFC 10030
0x1	Network Time Protocol Message	RFC 10030
0x2-0x7FFFFF	Unassigned	
0x800000-0xFFFFFE	Reserved for Experimental and Private Use	RFC 10030
0xFFFFFF	Reserved	RFC 10030

Table 1

Changes in the Specification Required range are approved by a designated expert (DE). The DE should be familiar with [RFC8126] (particularly Section 5 of [RFC8126]) and the current PTP specifications. The DE should verify that the specification of the organization-specific TLV identified by the assigned subtype exists and is publicly available. The purpose and use of the TLV should be sufficiently clear to enable interoperating implementations, without harming the protocol or the ecosystem.

4.2. NTP Extension Field Registration

IANA has allocated the following field in the "NTP Extension Field Types" registry <https://www.iana.org/assignments/ntp-parameters/> defined by [RFC5905]:

Field Type	Meaning	Reference
0x010A	Network Correction	RFC 10030

Table 2

5. Security Considerations

PTP transport prevents NTP clients from randomizing their source port as described in [RFC9109] because both requests and responses need to be sent to the PTP port in order to get a hardware receive timestamp and corrections from PTP transparent clocks.

The corrections provided by PTP transparent clocks cannot be authenticated. On-path attackers can modify the correction field, but only corrections smaller than the measured delay are accepted by clients. The impact is comparable to the impact of delaying unmodified NTP messages.

6. References

6.1. Normative References

[IEEE1588-2019] IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2019, DOI 10.1109/IEEESTD.2020.9120376, June 2020, <https://ieeexplore.ieee.org/document/9120376>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC7822] Mizrahi, T. and D. Mayer, "Network Time Protocol Version 4 (NTPv4) Extension Fields", RFC 7822, DOI 10.17487/RFC7822, March 2016, <https://www.rfc-editor.org/info/rfc7822>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

6.2. Informative References

[G8265-1] ITU-T, "Precision time protocol telecom profile for frequency synchronization", ITU-T Recommendation G.8265.1/Y.1365.1, November 2022, <https://www.itu.int/rec/T-REC-G.8265.1-202211-I/en>.

[G8275-2] ITU-T, "Precision time protocol telecom profile for phase/time synchronization with partial timing support from the network", ITU-T Recommendation G.8275.2/Y.1369.2, November 2022, <https://www.itu.int/rec/T-REC-G.8275.2-202211-I/en>.

[IEEE1588-2008] IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, DOI 10.1109/IEEESTD.2008.4579760, July 2008, <https://ieeexplore.ieee.org/document/4579760>.

[RFC5906] Haberman, B., Ed. and D. Mills, "Network Time Protocol Version 4: Autokey Specification", RFC 5906, DOI 10.17487/RFC5906, June 2010, <https://www.rfc-editor.org/info/rfc5906>.

[RFC8915] Franke, D., Sibold, D., Teichel, K., Dansarie, M., and R. Sundblad, "Network Time Security for the Network Time Protocol", RFC 8915, DOI 10.17487/RFC8915, September 2020, <https://www.rfc-editor.org/info/rfc8915>.

[RFC9109] Gont, F., Gont, G., and M. Lichvar, "Network Time Protocol Version 4: Port Randomization", RFC 9109, DOI 10.17487/RFC9109, August 2021, <https://www.rfc-editor.org/info/rfc9109>.

Acknowledgements

The author would like to thank Doug Arnold, Rodney Cummings, Martin Langer, and Robert Sparks for their comments and suggestions.

Author's Address

Miroslav Lichvar
Red Hat
Email: mlichvar@redhat.com