Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 8 March 2026

M. Lichvar
Red Hat

4 September 2025

NTP Over PTP
draft-ietf-ntp-over-ptp-05

Abstract

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 March 2026.

Copyright Notice

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 supported in all network devices involved in the synchronization (e.g. network interface controllers, switches, and routers) to eliminate the impact of software, processing and queueing delays on 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 and G.8275.2).

The Network Time Protocol [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 which 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 transport (e.g. UDP over IPv4, UDP over IPv6, 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 either as 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 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 which can timestamp only PTP messages and also 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 adds 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 IEEE1588 [IEEE1588-2019] are 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 like Network Time Security [RFC8915] are not impacted by the PTP transport. It is more difficult to secure PTP against delay attacks due to the sync message not being 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 which do not have full on-path PTP support, or where timestamping errors do not have a symmetric distribution (e.g. due to sensitivity to 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 transmit timestamping rate growing with the number of clients. A server which 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", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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

* type is 0x8000 (ORGANIZATION_EXTENSION_DO_NOT_PROPAGATE)

* lengthField is 8 + length of the NTP message

* organizationId is 00-00-5E (IANA OUI)

* organizationSubType is [[TBD]]

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

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.

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 send its response as the NTP TLV using the same PTP transport. The PTP message containing the NTP response MUST NOT be longer than the PTP message containing the NTP request. If the NTP response is expected to be used for synchronization (e.g. it is not an error message), the PTP message SHOULD have the same length as the message containing the request, using the PTP PAD TLV (type 0x8008) if needed, to avoid an asymmetric delay in networks without full on-path PTP support.

The PTP version 2.1 [IEEE1588-2019] specification states that "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 use the domainNumber and sdoId configured for use by NTP over PTP. The

domainNumber SHOULD be 123, and sdoId SHOULD be 0. This domainNumber 123 is not commonly used by PTP profiles, and so is less likely to interfere with any other PTP operation which might be running in the network.

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 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. 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. 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.

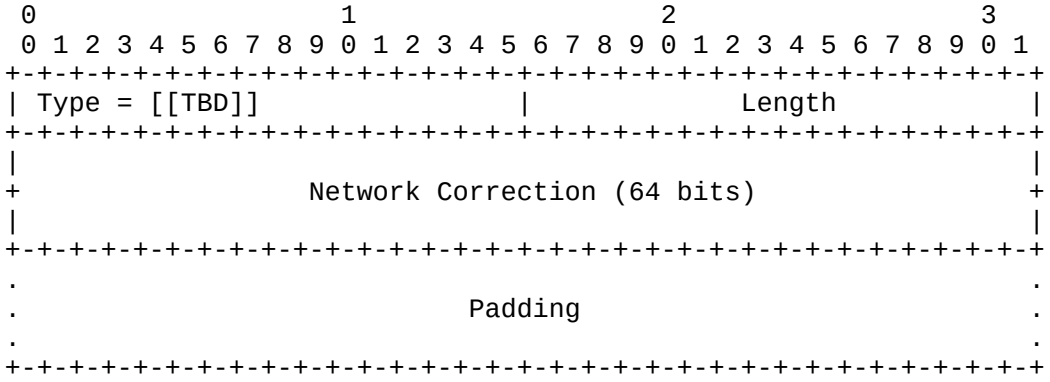

Figure 1: Format of Network Correction Extension Field

The length of the padding is the minimum required to make a valid extension field in the used version of NTP. In NTPv4 that is 16 octets to get a 28-octet extension field following RFC 7822 [RFC7822].

The Network Correction field in the extension field uses the 64-bit NTP timestamp format (resolution of about 1/4th of a nanosecond). The correction field in 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 of PTP receive timestamp corresponding to the beginning of the reception to NTP receive timestamp corresponding 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.

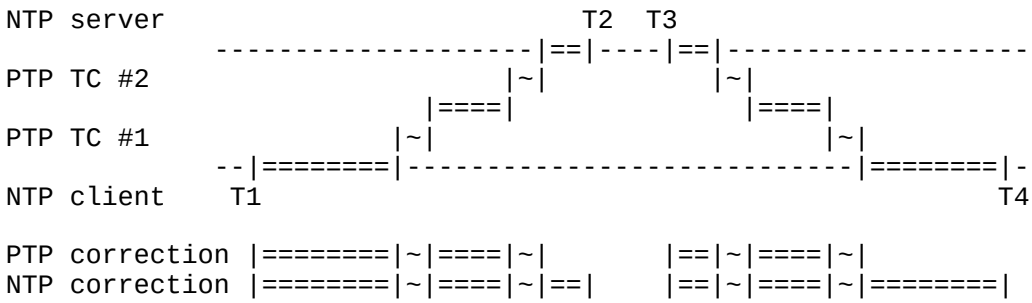

Figure 2: PTP vs NTP Correction

When an NTP server which 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 which 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 RFC 5905

* theta is the NTP offset from RFC 5905

* 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 man-in-the-middle attacks.

Root delay (DELTA) MUST NOT be corrected to not make the maximum assumed error (root distance) dependent on accurate network corrections.

The scaling by the freq_tc constant (e.g. 100 ppm) is needed to make room for errors in corrections made by transparent clocks running faster than true time and 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. Acknowledgements

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

5. IANA Considerations

IANA is requested to allocate the following field in the NTP Extension Field Types Registry [RFC5905]:

Field Type	Meaning	Reference
[[TBD]]	Network correction	[[this memo]]

Table 1

IANA is requested to create a new registry "IANA PTP TLV Subtypes Registry" for entries having the following fields:

Subtype (REQUIRED): An integer in the range 0-0xFFFFFF

Description (REQUIRED): A short text description

Reference (REQUIRED): 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

0x800000-0xFFFFFF: Reserved for Experimental and Private Use

The initial content of the registry is the following entry:

Subtype	Description	Reference
[[TBD]]	Network Time Protocol Message	[[this memo]]

Table 2

6. Implementation Status - RFC EDITOR: REMOVE BEFORE PUBLICATION

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 7942. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

6.1. chrony

chrony (https://chrony-project.org) added experimental support for NTP over PTP in version 4.2. As the type of the NTP TLV, it uses 0x2023 from the experimental "do not propagate" range.

It was tested on Linux with the following network controllers, which have hardware timestamping limited to PTP packets:

- Intel XL710 (i40e driver) - works
- Intel X540-AT2 (ixgbe driver) - works
- Intel 82576 (igb driver) - works
- Broadcom BCM5720 (tg3 driver) - works
- Broadcom BCM57810 (bnx2x driver) - does not timestamp unicast PTP packets
- Solarflare SFC9250 (sfc driver) - works

The network correction was tested with the following switches which support operation as a one-step E2E PTP unicast transparent clock:

- FS.COM IES3110-8TF-R - works
- Juniper QFX5200-32C-32Q - works

7. Security Considerations

The PTP transport prevents NTP clients from randomizing their source port.

The corrections provided by PTP transparent clocks cannot be authenticated. Man-in-the-middle 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.

8. References

8.1. Normative References

[IEEE1588-2019]
Institute of Electrical and Electronics Engineers, "IEEE std. 1588-2019, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems."", November 2019, <https://www.ieee.org>.

[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>.

8.2. Informative References

[IEEE1588-2008]
Institute of Electrical and Electronics Engineers, "IEEE std. 1588-2008, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems."", July 2008, <https://www.ieee.org>.

[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>.

Author’s Address

Miroslav Lichvar
Red Hat
Email: mlichvar@redhat.com