Network Working Group
Internet-Draft
Updates: 5905 (if approved)
Intended status: Standards Track
Expires: September 28, 2017

D. Franke
Akamai
A. Malhotra
Boston University

March 27, 2017

NTP Client Data Minimization
draft-dfranke-ntp-data-minimization-02

Abstract

This memo proposes backward-compatible updates to the Network Time Protocol to strip unnecessary identifying information from client requests and to improve resilience against blind spoofing of unauthenticated server responses.

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 http://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 September 28, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Requirements Language 2
3. Client Packet Format 2
4. Security and Privacy Considerations 3
4.1. Data Minimization 3
4.2. Transmit Timestamp Randomization 4
5. IANA Considerations 4
6. References 4
6.1. Normative References 4
6.2. Informative References 5
Appendix A. Acknowledgements 5
Authors' Addresses 5

1. Introduction

Network Time Protocol (NTP) packets, as specified by RFC 5905 [RFC5905], carry a great deal of information about the state of the NTP daemon which transmitted them. In the case of mode 4 packets (responses sent from server to client), as well as in broadcast (mode 5) and symmetric peering modes (mode 1/2), most of this information is essential for accurate and reliable time synchronizaton. However, in mode 3 packets (requests sent from client to server), most of these fields serve no purpose. Server implementations never need to inspect them, and they can achieve nothing by doing so. Populating these fields with accurate information is harmful to privacy of clients because it allows a passive observer to fingerprint clients and track them as they move across networks.

This memo updates RFC 5905 to redact unnecessary data from mode 3 packets. This is a fully backwards-compatible proposal. It calls for no changes on the server side, and clients which implement these updates will remain fully interoperable with existing servers.

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].

3. Client Packet Format

In every client-mode packet sent by a Network Time Protocol [RFC5905] implementation:

The first octet, which contains the leap indicator, version number, and mode fields, SHOULD be set to 0x23 (LI = 0, VN = 4, Mode = 3).

The Transmit Timestamp field SHOULD be set uniformly at random, generated by a mechanism suitable for cryptographic purposes. [RFC4086] provides guidance on the generation of random values.

The Poll field MAY be set to the actual polling interval as specified by RFC 5905, or else MAY be set to zero.

All other header fields, specifically the Stratum, Precision, Root Delay, Root Dispersion, Reference ID, Reference Timestamp, Origin Timestamp, and Receive Timestamp, SHOULD be set to zero.

Servers MUST allow client packets to conform to the above recommendations. This requirement shall not be construed so as to prohibit servers from rejecting conforming packets for unrelated reasons, such as access control or rate limiting.

4. Security and Privacy Considerations

4.1. Data Minimization

Zeroing out unused fields in client requests prevents disclosure of information that can be used for fingerprinting [RFC6973].

While populating any of these fields with authentic data reveals at least some identifying information about the client, the Origin Timestamp and Receive Timestamp fields constitute a particularly severe information leak. RFC 5905 calls for clients to copy the transmit timestamp and destination timestamp of the server’s most recent response into the origin timestamp and receive timestamp (respectively) of their next request to that server. Therefore, when a client moves between networks, a passive observer of both network paths can determine with high confidence that the old and new IP addresses belong to the same system by noticing that the transmit timestamp of a response sent to the old IP matches the origin timestamp of a request sent from the new one.

Zeroing the poll field is made optional (MAY rather than SHOULD) so as not to preclude future development of schemes wherein the server uses information about the client’s current poll interval in order to recommend adjustments back to the client. Putting accurate information into this field has no significant impact on privacy since an observer can already obtain this information simply by observing the actual interval between requests.

4.2. Transmit Timestamp Randomization

While this memo calls for most fields in client packets to be set to zero, the transmit timestamp is randomized. This decision is motivated by security as well as privacy.

NTP servers copy the transmit timestamp from the client's request into the origin timestamp of the response; this memo calls for no change in this behavior. Clients discard any response whose origin timestamp does not match the transmit timestamp of any request currently in flight.

In the absence of cryptographic authentication, verification of origin timestamps is clients' primary defense against blind spoofing of NTP responses. It is therefore important that clients' transmit timestamps be unpredictable. Their role in this regard is closely analagous to that of TCP Initial Sequence Numbers [RFC6528].

The traditional behavior of the NTP reference implementation is to randomize only a few (typically 10-15 depending on the precision of the system clock) low-order bits of transmit timestamp, with all higher bits representing the system time, as measured just before the packet was sent. This is suboptimal, because with so few random bits, an adversary sending spoofed packets at high volume will have a good chance of correctly guessing a valid origin timestamp.

5. IANA Considerations

[RFC EDITOR: DELETE PRIOR TO PUBLICATION]

This memo introduces no new IANA considerations.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://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, <http://www.rfc-editor.org/info/rfc5905>.

6.2. Informative References

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <http://www.rfc-editor.org/info/rfc4086>.

[RFC6528] Gont, F. and S. Bellovin, "Defending against Sequence Number Attacks", RFC 6528, DOI 10.17487/RFC6528, February 2012, <http://www.rfc-editor.org/info/rfc6528>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <http://www.rfc-editor.org/info/rfc6973>.

Appendix A. Acknowledgements

The authors thank Prof. Sharon Goldberg and Miroslav Lichvar for calling attention to the issues addressed in this memo.

Authors' Addresses

Daniel Fox Franke
Akamai Technologies, Inc.
150 Broadway
Cambridge, MA 02142
United States

Email: dafranke@akamai.com
URI: https://www.dfranke.us

Aanchal Malhotra
Boston University
111 Cummington St
Boston, MA 02215
United States

Email: aanchal4@bu.edu

Network Working Group
Internet-Draft
Intended status: Historic
Expires: April 20, 2017

D. Mills
University of Deleware
B. Haberman, Ed.
JHU
October 17, 2016

Control Messages Protocol for Use with Network Time Protocol Version 4
draft-haberman-ntpwg-mode-6-cmds-02

Abstract

This document describes the structure of the control messages used with the Network Time Protocol. These control messages can be used to monitor and control the Network Time Protocol application running on any IP network attached computer. The information in this document was originally described in Appendix B of RFC 1305. The goal of this document is to provide a historic description of the control messages.

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 http://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 April 20, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Control Message Overview 2
2. NTP Control Message Format 4
3. Status Words 5
3.1. System Status Word 6
3.2. Peer Status Word 8
3.3. Clock Status Word 9
3.4. Error Status Word 10
4. Commands 10
5. IANA Considerations 12
6. Security Considerations 12
7. Acknowledgements 14
8. Normative References 14
Authors' Addresses 14

1. Introduction

RFC 1305 [RFC1305] described a set of control messages for use within the Network Time Protocol (NTP) when a comprehensive network management solution was not available. The definitions of these control messages were not promulgated to RFC 5905 [RFC5905] when NTP version 4 was documented. These messages were intended for use only in systems where no other management facilities were available or appropriate, such as in dedicated-function bus peripherals. Support for these messages is not required in order to conform to RFC 5905 [RFC5905]. The control messages are described here as a historical record given their use within NTPv4.

1.1. Control Message Overview

The NTP Control Message has the value 6 specified in the mode field of the first octet of the NTP header and is formatted as shown in Figure 1. The format of the data field is specific to each command or response; however, in most cases the format is designed to be constructed and viewed by humans and so is coded in free-form ASCII. This facilitates the specification and implementation of simple management tools in the absence of fully evolved network-management facilities. As in ordinary NTP messages, the authenticator field follows the data field. If the authenticator is used the data field is zero-padded to a 32-bit boundary, but the padding bits are not considered part of the data field and are not included in the field count.

IP hosts are not required to reassemble datagrams larger than 576 octets; however, some commands or responses may involve more data than will fit into a single datagram. Accordingly, a simple reassembly feature is included in which each octet of the message data is numbered starting with zero. As each fragment is transmitted the number of its first octet is inserted in the offset field and the number of octets is inserted in the count field. The more-data (M) bit is set in all fragments except the last.

Most control functions involve sending a command and receiving a response, perhaps involving several fragments. The sender chooses a distinct, nonzero sequence number and sets the status field and R and E bits to zero. The responder interprets the opcode and additional information in the data field, updates the status field, sets the R bit to one and returns the three 32-bit words of the header along with additional information in the data field. In case of invalid message format or contents the responder inserts a code in the status field, sets the R and E bits to one and, optionally, inserts a diagnostic message in the data field.

Some commands read or write system variables and peer variables for an association identified in the command. Others read or write variables associated with a radio clock or other device directly connected to a source of primary synchronization information. To identify which type of variable and association a 16-bit association identifier is used. System variables are indicated by the identifier zero. As each association is mobilized a unique, nonzero identifier is created for it. These identifiers are used in a cyclic fashion, so that the chance of using an old identifier which matches a newly created association is remote. A management entity can request a list of current identifiers and subsequently use them to read and write variables for each association. An attempt to use an expired identifier results in an exception response, following which the list can be requested again.

Some exception events, such as when a peer becomes reachable or unreachable, occur spontaneously and are not necessarily associated with a command. An implementation may elect to save the event information for later retrieval or to send an asynchronous response (called a trap) or both. In case of a trap the IP address and port number is determined by a previous command and the sequence field is set as described below. Current status and summary information for the latest exception event is returned in all normal responses. Bits in the status field indicate whether an exception has occurred since the last response and whether more than one exception has occurred.

Commands need not necessarily be sent by an NTP peer, so ordinary access-control procedures may not apply; however, the optional mask/

match mechanism suggested elsewhere in this document provides the capability to control access by mode number, so this could be used to limit access for control messages (mode 6) to selected address ranges.

2. NTP Control Message Format

The format of the NTP Control Message header, which immediately follows the UDP header, is shown in Figure 1. Following is a description of its fields. Bit positions marked as zero are reserved and should always be transmitted as zero.

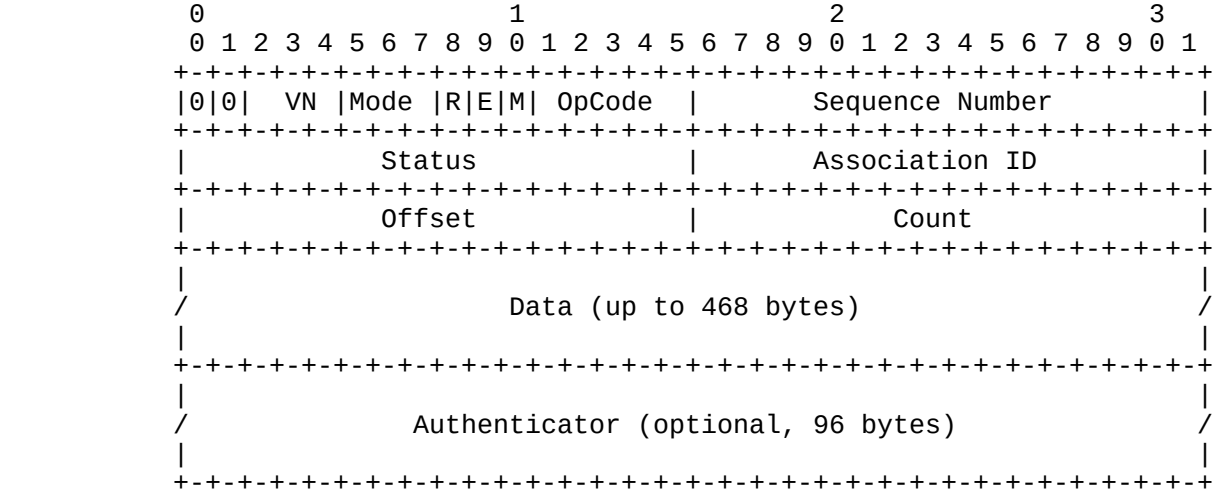


Figure 1: NTP Control Message Header

Version Number (VN): This is a three-bit integer indicating the NTP version number, currently four (4).

Mode: This is a three-bit integer indicating the mode. The value 6 indicates an NTP control message.

Response Bit (R): Set to zero for commands, one for responses.

Error Bit (E): Set to zero for normal response, one for error response.

More Bit (M): Set to zero for last fragment, one for all others.

Operation Code (OpCode): This is a five-bit integer specifying the command function. Values currently defined include the following:

Code	Meaning
0	reserved
1	read status command/response
2	read variables command/response
3	write variables command/response
4	read clock variables command/response
5	write clock variables command/response
6	set trap address/port command/response
7	trap response
8-31	reserved

Sequence Number: This is a 16-bit integer indicating the sequence number of the command or response.

Status: This is a 16-bit code indicating the current status of the system, peer or clock, with values coded as described in following sections.

Association ID: This is a 16-bit integer identifying a valid association.

Offset: This is a 16-bit integer indicating the offset, in octets, of the first octet in the data area.

Count: This is a 16-bit integer indicating the length of the data field, in octets.

Data: This contains the message data for the command or response. The maximum number of data octets is 468.

Authenticator (optional): When the NTP authentication mechanism is implemented, this contains the authenticator information defined in Appendix C of RFC 1305.

3. Status Words

Status words indicate the present status of the system, associations and clock. They are designed to be interpreted by network-monitoring programs and are in one of four 16-bit formats shown in Figure 2 and described in this section. System and peer status words are associated with responses for all commands except the read clock variables, write clock variables and set trap address/port commands. The association identifier zero specifies the system status word, while a nonzero identifier specifies a particular peer association. The status word returned in response to read clock variables and

write clock variables commands indicates the state of the clock hardware and decoding software. A special error status word is used to report malformed command fields or invalid values.

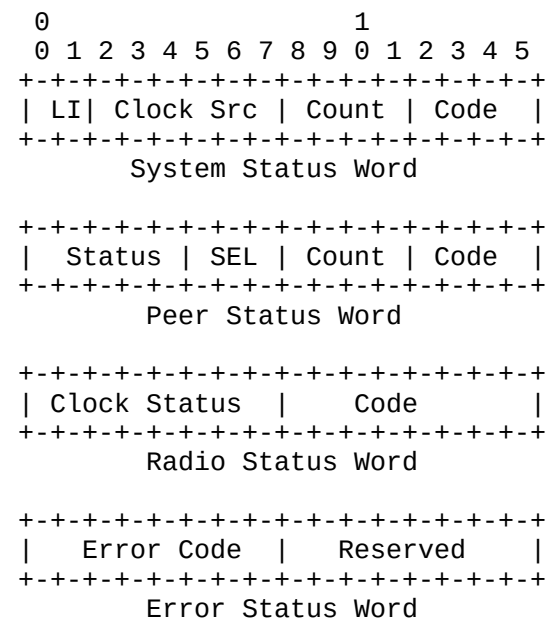


Figure 2: Status Word Formats

3.1. System Status Word

The system status word appears in the status field of the response to a read status or read variables command with a zero association identifier. The format of the system status word is as follows:

Leap Indicator (LI): This is a two-bit code warning of an impending leap second to be inserted/deleted in the last minute of the current day, with bit 0 and bit 1, respectively, coded as follows:

LI	Meaning
00	no warning
01	read status command/response
10	read variables command/response
11	write variables command/response

Clock Source (Clock Src): This is a six-bit integer indicating the current synchronization source, with values coded as follows:

Code	Meaning
0	unspecified or unknown
1	Calibrated atomic clock (e.g.,, HP 5061)
2	VLF (band 4) or LF (band 5) radio (e.g.,, OMEGA,, WWVB)
3	HF (band 7) radio (e.g.,, CHU,, MSF,, WWV/H)
4	UHF (band 9) satellite (e.g.,, GOES,, GPS)
5	local net (e.g.,, DCN,, TSP,, DTS)
6	UDP/NTP
7	UDP/TIME
8	eyeball-and-wristwatch
9	telephone modem (e.g.,, NIST)
10-63	reserved

System Event Counter (Count): This is a four-bit integer indicating the number of system exception events occurring since the last time the system status word was returned in a response or included in a trap message. The counter is cleared when returned in the status field of a response and freezes when it reaches the value 15.

System Event Code (Code): This is a four-bit integer identifying the latest system exception event, with new values overwriting previous values, and coded as follows:

Code	Meaning
0	unspecified
1	system restart
2	system or hardware fault
3	system new status word (leap bits or synchronization change)
4	system new synchronization source or stratum (sys.peer or sys.stratum change)
5	system clock reset (offset correction exceeds CLOCK.MAX)
6	system invalid time or date (see NTP specification)
7	system clock exception (see system clock status word)
8-15	reserved

3.2. Peer Status Word

A peer status word is returned in the status field of a response to a read status, read variables or write variables command and appears also in the list of association identifiers and status words returned by a read status command with a zero association identifier. The format of a peer status word is as follows:

Peer Status (Status): This is a five-bit code indicating the status of the peer determined by the packet procedure, with bits assigned as follows:

Peer Status	Meaning
0	configured (peer.config)
1	authentication enabled (peer.authenable)
2	authentication okay (peer.authentic)
3	reachability okay (peer.reach <F128M>?F255D> 0)
4	reserved

Peer Selection (SEL): This is a three-bit integer indicating the status of the peer determined by the clock-selection procedure, with values coded as follows:

Sel	Meaning
0	rejected
1	passed receive sanity checks
2	passed correctness check (intersection algorithm
3	passed candidate checks (if limit check implemented)
4	passed outlyer checks (cluster algorithm
5	current synchronization source; max distance exceeded (if limit check implemented)
6	current synchronization source; max distance okay
7	reserved

Peer Event Counter (Count): This is a four-bit integer indicating the number of peer exception events that occurred since the last time the peer status word was returned in a response or included in a trap message. The counter is cleared when returned in the status field of a response and freezes when it reaches the value 15.

Peer Event Code (Code): This is a four-bit integer identifying the latest peer exception event, with new values overwriting previous values, and coded as follows:

Peer Event Code	Meaning
0	unspecified
1	peer IP error
2	peer authentication failure (peer.authentic bit 1 --> 0
3	peer unreachable (peer.reach was nonzero now zero)
4	peer reachable (peer.reach was zero now nonzero)
5	peer clock exception (see peer clock status word)
6-15	reserved

3.3. Clock Status Word

There are two ways a reference clock can be attached to a NTP service host, as an dedicated device managed by the operating system and as a synthetic peer managed by NTP. As in the read status command, the association identifier is used to identify which one, zero for the system clock and nonzero for a peer clock. Only one system clock is supported by the protocol, although many peer clocks can be supported. A system or peer clock status word appears in the status field of the response to a read clock variables or write clock variables command. This word can be considered an extension of the system status word or the peer status word as appropriate. The format of the clock status word is as follows:

Clock Status: This is an eight-bit integer indicating the current clock status, with values coded as follows:

Clock Status	Meaning
0	clock operating within nominals
1	reply timeout
2	bad reply format
3	hardware or software fault
4	propagation failure
5	bad date format or value
6	bad time format or value
7-255	reserved

Clock Event Code (Code): This is an eight-bit integer identifying the latest clock exception event, with new values overwriting previous values. When a change to any nonzero value occurs in the radio status field, the radio status field is copied to the clock event code field and a system or peer clock exception event is declared as appropriate.

3.4. Error Status Word

An error status word is returned in the status field of an error response as the result of invalid message format or contents. Its presence is indicated when the E (error) bit is set along with the response (R) bit in the response. It consists of an eight-bit integer coded as follows:

Error Status	Meaning
0	unspecified
1	authentication failure
2	invalid message length or format
3	invalid opcode
4	unknown association identifier
5	unknown variable name
6	invalid variable value
7	administratively prohibited
8-255	reserved

4. Commands

Commands consist of the header and optional data field shown in Figure 2. When present, the data field contains a list of identifiers or assignments in the form <<identifier>>[=<<value>>],<<identifier>>[=<<value>>],... where <<identifier>> is the ASCII name of a system or peer variable specified in RFC 5905 and <<value>> is expressed as a decimal, hexadecimal or string constant in the syntax of the C programming language. Where no ambiguity exists, the <169>sys.<170> or <169>peer.<170> prefixes can be suppressed. Whitespace (ASCII nonprinting format effectors) can be added to improve readability for simple monitoring programs that do not reformat the data field. Internet addresses are represented as four octets in the form [n.n.n.n], where n is in decimal notation and the brackets are optional. Timestamps, including reference, originate, receive and transmit values, as well as the logical clock, are represented in units of seconds and fractions, preferably in hexadecimal notation, while delay, offset, dispersion and distance values are represented

in units of milliseconds and fractions, preferably in decimal notation. All other values are represented as-is, preferably in decimal notation.

Implementations may define variables other than those described in RFC 5905. Called extramural variables, these are distinguished by the inclusion of some character type other than alphanumeric or <169>.<170> in the name. For those commands that return a list of assignments in the response data field, if the command data field is empty, it is expected that all available variables defined in RFC 5905 will be included in the response. For the read commands, if the command data field is nonempty, an implementation may choose to process this field to individually select which variables are to be returned.

Commands are interpreted as follows:

Read Status (1): The command data field is empty or contains a list of identifiers separated by commas. The command operates in two ways depending on the value of the association identifier. If this identifier is nonzero, the response includes the peer identifier and status word. Optionally, the response data field may contain other information, such as described in the Read Variables command. If the association identifier is zero, the response includes the system identifier (0) and status word, while the data field contains a list of binary-coded pairs <<association identifier>> <<status word>>, one for each currently defined association.

Read Variables (2): The command data field is empty or contains a list of identifiers separated by commas. If the association identifier is nonzero, the response includes the requested peer identifier and status word, while the data field contains a list of peer variables and values as described above. If the association identifier is zero, the data field contains a list of system variables and values. If a peer has been selected as the synchronization source, the response includes the peer identifier and status word; otherwise, the response includes the system identifier (0) and status word.

Write Variables (3): The command data field contains a list of assignments as described above. The variables are updated as indicated. The response is as described for the Read Variables command.

Read Clock Variables (4): The command data field is empty or contains a list of identifiers separated by commas. The association identifier selects the system clock variables or peer clock variables in the same way as in the Read Variables command. The response

includes the requested clock identifier and status word and the data field contains a list of clock variables and values, including the last timecode message received from the clock.

Write Clock Variables (5): The command data field contains a list of assignments as described above. The clock variables are updated as indicated. The response is as described for the Read Clock Variables command.

Set Trap Address/Port (6): The command association identifier, status and data fields are ignored. The address and port number for subsequent trap messages are taken from the source address and port of the control message itself. The initial trap counter for trap response messages is taken from the sequence field of the command. The response association identifier, status and data fields are not significant. Implementations should include sanity timeouts which prevent trap transmissions if the monitoring program does not renew this information after a lengthy interval.

Trap Response (7): This message is sent when a system, peer or clock exception event occurs. The opcode field is 7 and the R bit is set. The trap counter is incremented by one for each trap sent and the sequence field set to that value. The trap message is sent using the IP address and port fields established by the set trap address/port command. If a system trap the association identifier field is set to zero and the status field contains the system status word. If a peer trap the association identifier field is set to that peer and the status field contains the peer status word. Optional ASCII-coded information can be included in the data field.

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. Security Considerations

A number of security vulnerabilities have been identified with these control messages.

NTP’s control query interface allows reading and writing of system, peer, and clock variables remotely from arbitrary IP addresses using commands mentioned in Section 4. Traditionally, overwriting these variables, but not reading them, requires authentication by default. However, this document argues that an NTP host must authenticate all control queries and not just ones that overwrite these variables.

Alternatively, the host can use a whitelist to explicitly list IP addresses that are allowed to control query the clients. These access controls are required for the following reasons:

- NTP as a Distributed Denial-of-Service (DDoS) vector. NTP timing query and response packets (modes 1-2, 3-4, 5) are usually short in size. However, some NTP control queries generate a very long packet in response to a short query. As such, there is a history of use of NTP's control queries, which exhibit such behavior, to perform DDoS attacks. These off-path attacks exploit the large size of NTP control queries to cause UDP-based amplification attacks (e.g., mode 7 monlist command generates a very long packet in response to a small query (CVE-2013-5211)). These attacks only use NTP as a vector for DoS atacks on other protocols, but do not affect the time service on the NTP host itself.

- Time-shifting attacks through information leakage/overwriting. NTP hosts save important system and peer state variables. An off-path attacker who can read these variables remotely can leverage the information leaked by these control queries to perform time-shifting and DoS attacks on NTP clients. These attacks do affect time synchronization on the NTP hosts. For instance,

  * In the client/server mode, the client stores its local time when it sends the query to the server in its xmt peer variable. This variable is used to perform TEST2 to non-cryptographically authenticate the server, i.e., if the origin timestamp field in the corresponding server response packet matches the xmt peer variable, then the client accepts the packet. An off-path attacker, with the ability to read this variable can easily spoof server response packets for the client, which will pass TEST2, and can deny service or shift time on the NTP client. CVE-2015-8139 describes the specific attack.

  * The client also stores its local time when the server response is received in its rec peer variable. This variable is used for authentication in interleaved-pivot mode. An off-path attacker with the ability to read this state variable can easily shift time on the client by passing this test. CVE-2016-1548 describes the attack.

- Fast-Scanning. NTP mode 6 control messages are usually small UDP packets. Fast-scanning tools like ZMap can be used to spray the entire (potentially reachable) Internet with these messages within hours to identify vulnerable hosts. To make things worse, these attacks can be extremely low-rate, only requiring a control query for reconnaissance and a spoofed response to shift time on vulnerable clients. CVE-2016-1548 is one such example.

NTP best practices recommend configuring ntpd with the no-query parameter. The no-query parameter blocks access to all remote control queries. However, sometimes the nosts do not want to block all queries and want to give access for certain control queries remotely. This could be for the purpose of remote management and configuration of the hosts in certain scenarios. Such hosts tend to use firewalls or other middleboxes to blacklist certain queries within the network.

Recent work (reference needed) shows that significantly fewer hosts respond to mode 7 monlist queries as compared to other control queries because it is a well-known and exploited control query. These queries are likely blocked using blacklists on firewalls and middleboxes rather than the no-query option on NTP hosts. The remaining control queries that can be exploited likely remain out of the blacklist because they are undocumented in the current NTP specification [RFC5905].

This document describes all of the mode 6 control queries allowed by NTP and can help administrators make informed decisions on security measures to protect NTP devices from harmful queries and likely make those systems less vulnerable.

7. Acknowledgements

Tim Plunkett created the original version of this document. Aanchal Malhotra provided the initial version of the Security Considerations section.

8. Normative References

[RFC1305] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", RFC 1305, DOI 10.17487/RFC1305, March 1992, <http://www.rfc-editor.org/info/rfc1305>.

[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, <http://www.rfc-editor.org/info/rfc5905>.

Authors' Addresses

Dr. David L. Mills
University of Deleware

Email: mills@udel.edu

Brian Haberman (editor)
JHU

Email: brian@innovationslab.net

Internet Engineering Task Force
Internet-Draft
Updates: 5905 (if approved)
Intended status: Standards Track
Expires: July 8, 2019

A. Malhotra
S. Goldberg
Boston University

January 4, 2019

Message Authentication Code for the Network Time Protocol
draft-ietf-ntp-mac-06

Abstract

RFC 5905 states that Network Time Protocol (NTP) packets should be authenticated by appending the NTP data to a 128-bit key, and hashing the result with MD5 to obtain a 128-bit tag. This document deprecates MD5-based authentication, which is considered to be too weak, and recommends the use of AES-CMAC as in RFC 4493 as a replacement.

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 July 8, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Requirements Language 2
2. Deprecating the use of MD5 2
3. Replacement Recommendation 3
4. Motivation 3
5. Test Vectors 3
6. Security Considerations 3
7. Acknowledgements 4
8. IANA Considerations 4
9. References 4
9.1. Normative References 4
9.2. Informative References 4
Authors' Addresses 5

1. Introduction

RFC 5905 [RFC5905] states that Network Time Protocol (NTP) packets should be authenticated by appending the NTP data to a 128-bit key, and hashing the result with MD5 to obtain a 128-bit tag. This document deprecates MD5-based authentication, which is considered to be too weak, and recommends the use of AES-CMAC [RFC4493] as a replacement.

1.1. 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. Deprecating the use of MD5

RFC 5905 [RFC5905] defines how the MD5 digest algorithm in RFC 1321 [RFC1321] can be used as a message authentication code (MAC) for authenticating NTP packets. However, as discussed in [BCK] and RFC 6151 [RFC6151], this is not a secure MAC and therefore MUST be deprecated.

3. Replacement Recommendation

If NTP authentication is implemented, then AES-CMAC as specified in RFC 4493 [RFC4493] MUST be computed over all fields in the NTP header, and any extension fields that are present in the NTP packet as described in RFC 5905 [RFC5905]. The MAC key for NTP MUST be 128 bits long AES-128 key and the resulting MAC tag MUST be at least 128 bits long as stated in section 2.4 of RFC 4493 [RFC4493]. NTP makes this transition possible as it supports algorithm agility as described in Section 2.1 of RFC 7696 [RFC7696].

The hosts who wish to use NTP authentication share a symmetric key out-of-band. So they MUST implement AES-CMAC and share the corresponding symmetric key. A symmetric key is a triplet of ID, type (e.g. MD5, AES-CMAC) and the key itself. All three have to match in order to succesfully authenticate packets between two hosts. Old implementations that don't support AES-CMAC will not accept and will not send packets authenticated with such a key.

4. Motivation

AES-CMAC is recommended for the following reasons:

1. It is an IETF standard that is available in many open source implementations.

2. It is immune to nonce-reuse vulnerabilities (e.g. [Joux]) because it does not use a nonce.

3. It has fine performance in terms of latency and throughput.

4. It benefits from native hardware support, for instance, Intel's New Instruction set GUE [GUE].

5. Test Vectors

For test vectors and their outputs refer to Section 4 of RFC 4493 [RFC4493]

6. Security Considerations

Refer to the Appendices A, B and C of NIST document on recommendation for the CMAC mode of authentication [NIST] and Security Considerations Section of RFC 4493 [RFC4493] for discussion on security guarantees of AES-CMAC.

7. Acknowledgements

The authors wish to acknowledge useful discussions with Leen Alshenibr, Daniel Franke, Ethan Heilman, Kenny Paterson, Leonid Reyzin, Harlan Stenn, and Mayank Varia.

8. IANA Considerations

This memo includes no request to IANA.

9. References

9.1. Normative References

[NIST] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication", <https://www.nist.gov/publications/recommendation-block-cipher-modes-operation-cmac-mode-authentication-0>.

[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>.

[RFC4493] Song, JH., Poovendran, R., Lee, J., and T. Iwata, "The AES-CMAC Algorithm", RFC 4493, DOI 10.17487/RFC4493, June 2006, <https://www.rfc-editor.org/info/rfc4493>.

[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>.

9.2. Informative References

[BCK] Bellare, M., Canetti, R., and H. Krawczyk, "Keyed Hash Functions and Message Authentication", in Proceedings of Crypto'96, 1996.

[GUE] Geuron, S., "Intel Advanced Encryption Standard (AES) New Instructions Set", <https://www.intel.com/content/dam/doc/white-paper/advanced-encryption-standard-new-instructions-set-paper.pdf>.

[Joux] Joux, A., "Authentication Failures in NIST version of GCM", <http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/800-38_Series-Drafts/GCM/Joux_comments.pdf>.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <https://www.rfc-editor.org/info/rfc1321>.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <https://www.rfc-editor.org/info/rfc6151>.

[RFC7696] Housley, R., "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", BCP 201, RFC 7696, DOI 10.17487/RFC7696, November 2015, <https://www.rfc-editor.org/info/rfc7696>.

[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>.

Authors' Addresses

Aanchal Malhotra
Boston University
111 Cummington St
Boston, MA 02215
US

Email: aanchal4@bu.edu

Sharon Goldberg
Boston University
111 Cummington St
Boston, MA 02215
US

Email: goldbe@cs.bu.edu

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 7, 2019

H. Stenn
Network Time Foundation
S. Goldberg
Boston University
October 4, 2018

Network Time Protocol REFID Updates
draft-ietf-ntp-refid-updates-04

Abstract

RFC 5905 [RFC5905], section 7.3, "Packet Header Variables", defines the value of the REFID, the system peer for the responding host. In the past, for IPv4 associations the IPv4 address is used, and for IPv6 associations the first four octets of the MD5 hash of the IPv6 are used. There are two recognized shortcomings to this approach, and this proposal addresses them. One is that knowledge of the system peer is "abusable" information and should not be generally available. The second is that the four octet hash of the IPv6 address looks very much like an IPv4 address, and this is confusing.

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 April 7, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. The REFID 2
1.2. NOT-YOU REFID 3
1.3. IPv6 REFID 3
1.4. Requirements Language 4
2. The NOT-YOU REFID 4
2.1. Proposal 4
3. Augmenting the IPv6 REFID Hash 5
3.1. Background 5
3.2. Potential Problems 6
3.3. Questions 6
4. Acknowledgements 6
5. IANA Considerations 6
6. Security Considerations 6
7. References 7
7.1. Normative References 7
7.2. Informative References 7
Authors' Addresses 7

1. Introduction

1.1. The REFID

The interpretation of a REFID is based on the stratum, as documented in RFC 5905 [RFC5905], section 7.3, "Packet Header Variables". The core reason for the REFID in the NTP Protocol is to prevent a degree-one timing loop, where server B decides to follow A as its time source, and A then decides to follow B as its time source.

At Stratum 2+, which will be the case if two servers A and B are exchanging timing information, then if server B follows A as its time source, A's address will be B's REFID. When A uses IPv4, the default REFID is A's IPv4 address. When A uses IPv6, the default REFID is a four-octet digest of A's IPv6 address. Now, if A queries B for its time, then A will learn that B is using A as its time source by observing A's address in the REFID field of the response packet sent by B. Thus, A will not select B as a potential time source, as this would cause a timing loop.

1.2. NOT-YOU REFID

The traditional REFID mechanism, however, also allows a third-party C to learn that A is the time source that is being used by B. When A is using IPv4, C can learn this by querying B for its time, and observing that the REFID in B's response is the IPv4 address of A. Meanwhile, when A is using IPv6, then C can again query B for its time, and then can use an offline dictionary attack to attempt to determine the IPv6 address that corresponds to the digest value in the response sent by B. C could construct the necessary dictionary by compiling a list of publicly accessible IPv6 servers. Remote attackers can use this technique to attempt to identify the time sources used by a target, and then send spoofed packets to the target or its time source in an attempt to disrupt time service, as was done e.g., in [NDSS16] or [CVE-2015-8138].

The REFID thus unnecessarily leaks information about a target's time server to remote attackers. The best way to mitigate this vulnerability is to decouple the IP address of the time source from the REFID. To do this, a system can use an otherwise-impossible value for its REFID, called the "not-you" value, when it believes that a querying system is not its time source.

The NOT-YOU REFID proposal is backwards-compatible. It can be implemented by one peer in an NTP association without any changes to the other peer.

The NOT-YOU REFID proposal does have a small risk, in that a system that might return NOT-YOU does not have perfect information and it is possible that the remote system peer is contacting "us" via a different network interface. In this case, the remote system might choose us as their system peer, and a degree-one timing loop will occur. In this case, however, the two systems will spiral into worsening stratum positions with increasing root distances, and eventually the loop will break. If any other systems are available as time servers, one of them may become the new system peer. However, unless or until this happens the two spiraling systems will have degraded time quality.

1.3. IPv6 REFID

In an environment where all time queries made to a server can be trusted, an operator might well choose to expose the real REFID. RFC 5905 [RFC5905], section 7.3, "Packet Header Variables", explains how a remote system peer is converted to a REFID. It says:

> If using the IPv4 address family, the identifier is the four-octet IPv4 address. If using the IPv6 family, it is the first four octets of the MD5 hash of the IPv6 address. ...

However, the MD5 hash of an IPv6 address often looks like a valid IPv4 address. When this happens, an operator cannot tell if the REFID refers to an IPv6 address or and IPv4. Specifically, the NTP Project has received a report where the generated IPv6 hash decoded to the IPv4 address of a different machine on the system peer's network.

This proposal offers a way for a system to generate a REFID for a IPv6 system peer that does not conflict with an IPv4-based REFID.

This proposal is not fully backwards-compatible. It SHOULD be implemented by both peers in an NTP association. In the scenario where A and B are peering using IPv6, where A is the system peer and does not understand IPv6 REFID, and B is subordinate and is using IPv6 REFID, A will not be able to determine that B is using A as its system peer and a degree-one timing loop can form.

If both peers implement the IPv6 REFID this situation cannot happen.

[If at least one of the peers implements the proposed I-DO protocol this situation cannot happen.]

1.4. 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. The NOT-YOU REFID

2.1. Proposal

When enabled, this proposal allows the one-degree loop detection to work and useful diagnostic information to be provided to trusted partners while keeping potentially abusable information from being disclosed to ostensibly uninterested parties. It does this by returning the normal REFID to queries that come from trusted addresses or from an address that the current system believes is its time source (aka its "system peer"), and otherwise returning one of two special IP addresses that is interpreted to mean "not you". The "not you" IP addresses are 127.127.127.127 and 127.127.127.128. If an IPv6 query is received from an address whose four-octet hash equals one of these two addresses and we believe the querying host is

not our system peer, the other NOT-YOU address is returned as the REFID.

This mechanism is correct and transparent when the system responding with a NOT-YOU can accurately detect when it's getting a timing query from its system peer. A querying system that uses IPv4 continues to check that its IPv4 address does not appear in the REFID before deciding whether to take time from the current system. A querying system that uses IPv6 continues to check that the four-octet hash of its IPv6 address does not appear in the REFID before deciding whether to take time from the current system. However...

Use of the NOT-YOU REFID proposal will hide the current system's system peer from querying systems that the current system believes are not the current system's system peer. Should the current system return the "not you" REFID to a query from its system peer, for example in the case where the system peer sends its query from an unexpected IP address, a one-degree timing loop can occur. Put another way, the responding system has imperfect knowledge about whether or not the sender is its system peer and there are cases where it will offer a NOT-YOU response to its system peer, which can then produce a degree-one timing loop.

Note that this mechanism fully supports degree-one loop detection in the case where the responding NOT-YOU system can accurately detect when it's getting a request from its system peer, and otherwise provides the most basic diagnostic information to third parties.

3. Augmenting the IPv6 REFID Hash

3.1. Background

In a trusted network, the S2+ REFID is generated based on the network system peer. RFC 5905 [RFC5905] says:

> If using the IPv4 address family, the identifier is the four-octet IPv4 address. If using the IPv6 family, it is the first four octets of the MD5 hash of the IPv6 address. ...

This means that the IPv4 representation of the IPv6 hash would be: b1.b2.b3.b4 . This proposal is that the system MAY also use 255.b2.b3.b4 as its REFID. This reduces the risk of ambiguity, since addresses beginning with 255 are "reserved", and thus will not collide with valid IPv4 on the network.

When using the REFID to check for a timing loop for an IPv6 association, if the code that checks the first four-octets of the

hash fails to match then the code must check again, using 0xFF as the first octet of the hash.

3.2. Potential Problems

There is a 1 in 16,777,216 chance that the REFID hashes of two IPv6 addresses will be identical, producing a false-positive loop detection. With a sufficient number of servers, the risk of this problem becomes a non-issue. [The use of the NOT-YOU REFID and/or the proposed "REFID Suggestion" or "I-DO" extension fields are ways to mitigate this potential situation.]

Unrealistically, if only two instances of NTP are communicating via IPv6 and system A implements this new IPv6 REFID hash and system B does not, system B will not be able to detect this loop condition. In this case, the two machines will slowly increase their Stratum until they reach S16 and become unsynchronized. This situation is considered to be unrealistic because, for this to happen, each system would have to have only the other system available as a time source, for example, in a misconfigured "orphan mode" setup. There is no risk of this happening in an NTP network with 3 or more time sources, or in a properly-configured "time island" setup.

3.3. Questions

Should we reference the REFID Suggestion and I-DO proposals here?

4. Acknowledgements

For the "not-you" REFID, we acknowledge useful discussions with Aanchal Malhotra and Matthew Van Gundy.

For the IPv6 REFID, we acknowledge Dan Mahoney (and perhaps others) for suggesting the idea of using an "impossible" first-octet value to indicate an IPv6 refid hash.

5. IANA Considerations

This memo requests IANA to allocate a pseudo Extension Field Type of 0xFFFF so the proposed "I-Do" exchange can report whether or not the "IPv6 REFID Hash" is supported.

6. Security Considerations

Many systems running NTP are configured to return responses to timing queries by default. These responses contain a REFID field, which generally reveals the address of the system’s time source if that source is an IPv4 address. This behavior can be exploited by remote

attackers who wish to first learn the address of a target's time source, and then attack the target and/or its time source. As such, the "not-you" REFID proposal is designed to harden NTP against these attacks by limiting the amount of information leaked in the REFID field.

Systems running NTP should reveal the identity of their system in peer in their REFID only when they are on a trusted network. The IPv6 REFID proposal provides one way to do this, when the system peer uses addresses in the IPv6 family.

7. References

7.1. Normative References

[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>.

7.2. Informative References

[CVE-2015-8138] Van Gundy, M. and J. Gardner, "Network Time Protocol Origin Timestamp Check Impersonation Vulnerability (CVE-2015-8138)", in TALOS VULNERABILITY REPORT (TALOS-2016-0077), 2016.

[NDSS16] Malhotra, A., Cohen, I., Brakke, E., and S. Goldberg, "Attacking the Network Time Protocol", in ISOC Network and Distributed System Security Symposium 2016 (NDSS'16), 2016.

Authors' Addresses

Harlan Stenn
Network Time Foundation
P.O. Box 918
Talent, OR 97540
US

Email: stenn@nwtime.org

Sharon Goldberg
Boston University
111 Cummington St
Boston, MA 02215
US

Email: goldbe@cs.bu.edu

NTP Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 17, 2019

D. Franke
Akamai
D. Sibold
K. Teichel
PTB
M. Dansarie

R. Sundblad
Netnod

February 13, 2019

Network Time Security for the Network Time Protocol
draft-ietf-ntp-using-nts-for-ntp-17

Abstract

This memo specifies Network Time Security (NTS), a mechanism for using Transport Layer Security (TLS) and Authenticated Encryption with Associated Data (AEAD) to provide cryptographic security for the client-server mode of the Network Time Protocol (NTP).

NTS is structured as a suite of two loosely coupled sub-protocols. The first (NTS-KE) handles initial authentication and key establishment over TLS. The second handles encryption and authentication during NTP time synchronization via extension fields in the NTP packets, and holds all required state only on the client via opaque cookies.

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 August 17, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
1.1. Objectives 3
1.2. Protocol Overview 4
2. Requirements Language 7
3. TLS profile for Network Time Security 7
4. The NTS Key Establishment Protocol 7
4.1. NTS-KE Record Types 9
4.1.1. End of Message 9
4.1.2. NTS Next Protocol Negotiation 10
4.1.3. Error 10
4.1.4. Warning 10
4.1.5. AEAD Algorithm Negotiation 11
4.1.6. New Cookie for NTPv4 11
4.1.7. NTPv4 Server Negotiation 12
4.1.8. NTPv4 Port Negotiation 12
4.2. Key Extraction (generally) 13
5. NTS Extension Fields for NTPv4 13
5.1. Key Extraction (for NTPv4) 13
5.2. Packet Structure Overview 14
5.3. The Unique Identifier Extension Field 14
5.4. The NTS Cookie Extension Field 15
5.5. The NTS Cookie Placeholder Extension Field 15
5.6. The NTS Authenticator and Encrypted Extension Fields Extension Field 15
5.7. Protocol Details 17
6. Suggested Format for NTS Cookies 22
7. IANA Considerations 23
7.1. Service Name and Transport Protocol Port Number Registry 23
7.2. TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs Registry 23
7.3. TLS Exporter Labels Registry 24

7.4. NTP Kiss-o'-Death Codes Registry 24
7.5. NTP Extension Field Types Registry 24
7.6. Network Time Security Key Establishment Record Types Registry 25
7.7. Network Time Security Next Protocols Registry 26
7.8. Network Time Security Error and Warning Codes Registries 27
8. Implementation Status - RFC EDITOR: REMOVE BEFORE PUBLICATION 28
8.1. Implementation PoC 1 28
8.1.1. Coverage 28
8.1.2. Licensing 29
8.1.3. Contact Information 29
8.1.4. Last Update 29
8.2. Implementation PoC 2 29
8.2.1. Coverage 29
8.2.2. Licensing 29
8.2.3. Contact Information 29
8.2.4. Last Update 29
8.3. Interoperability 30
9. Security Considerations 30
9.1. Sensitivity to DDoS attacks 30
9.2. Avoiding DDoS Amplification 30
9.3. Initial Verification of Server Certificates 31
9.4. Delay Attacks 32
9.5. Random Number Generation 33
9.6. NTS Stripping 33
10. Privacy Considerations 33
10.1. Unlinkability 33
10.2. Confidentiality 34
11. Acknowledgements 34
12. References 35
12.1. Normative References 35
12.2. Informative References 36
Appendix A. Terms and Abbreviations 37
Authors' Addresses 38

1. Introduction

This memo specifies Network Time Security (NTS), a cryptographic security mechanism for network time synchronization. A complete specification is provided for application of NTS to the client-server mode of the Network Time Protocol (NTP) [RFC5905].

1.1. Objectives

The objectives of NTS are as follows:

- Identity: Through the use of the X.509 public key infrastructure, implementations may cryptographically establish the identity of the parties they are communicating with.

- Authentication: Implementations may cryptographically verify that any time synchronization packets are authentic, i.e., that they were produced by an identified party and have not been modified in transit.

- Confidentiality: Although basic time synchronization data is considered non-confidential and sent in the clear, NTS includes support for encrypting NTP extension fields.

- Replay prevention: Client implementations may detect when a received time synchronization packet is a replay of a previous packet.

- Request-response consistency: Client implementations may verify that a time synchronization packet received from a server was sent in response to a particular request from the client.

- Unlinkability: For mobile clients, NTS will not leak any information additional to NTP which would permit a passive adversary to determine that two packets sent over different networks came from the same client.

- Non-amplification: Implementations (especially server implementations) may avoid acting as distributed denial-of-service (DDoS) amplifiers by never responding to a request with a packet larger than the request packet.

- Scalability: Server implementations may serve large numbers of clients without having to retain any client-specific state.

1.2. Protocol Overview

The Network Time Protocol includes many different operating modes to support various network topologies. In addition to its best-known and most-widely-used client-server mode, it also includes modes for synchronization between symmetric peers, a control mode for server monitoring and administration, and a broadcast mode. These various modes have differing and partly contradictory requirements for security and performance. Symmetric and control modes demand mutual authentication and mutual replay protection. Additionally, for certain message types control mode may require confidentiality as well as authentication. Client-server mode places more stringent requirements on resource utilization than other modes, because servers may have vast number of clients and be unable to afford to

maintain per-client state. However, client-server mode also has more relaxed security needs, because only the client requires replay protection: it is harmless for stateless servers to process replayed packets. The security demands of symmetric and control modes, on the other hand, are in conflict with the resource-utilization demands of client-server mode: any scheme which provides replay protection inherently involves maintaining some state to keep track of what messages have already been seen.

This memo specifies NTS exclusively for the client-server mode of NTP. To this end, NTS is structured as a suite of two protocols:

The "NTS Extensions for NTPv4" define a collection of NTP extension fields for cryptographically securing NTPv4 using previously-established key material. They are suitable for securing client-server mode because the server can implement them without retaining per-client state. All state is kept by the client and provided to the server in the form of an encrypted cookie supplied with each request. On the other hand, the NTS Extension Fields are suitable *only* for client-server mode because only the client, and not the server, is protected from replay.

The "NTS Key Establishment" protocol (NTS-KE) is a mechanism for establishing key material for use with the NTS Extension Fields for NTPv4. It uses TLS to exchange keys, provide the client with an initial supply of cookies, and negotiate some additional protocol options. After this exchange, the TLS channel is closed with no per-client state remaining on the server side.

The typical protocol flow is as follows: The client connects to an NTS-KE server on the NTS TCP port and the two parties perform a TLS handshake. Via the TLS channel, the parties negotiate some additional protocol parameters and the server sends the client a supply of cookies along with a list of one or more IP addresses to NTP servers for which the cookies are valid. The parties use TLS key export [RFC5705] to extract key material which will be used in the next phase of the protocol. This negotiation takes only a single round trip, after which the server closes the connection and discards all associated state. At this point the NTS-KE phase of the protocol is complete. Ideally, the client never needs to connect to the NTS-KE server again.

Time synchronization proceeds with one of the indicated NTP servers over the NTP UDP port. The client sends the server an NTP client packet which includes several extension fields. Included among these fields are a cookie (previously provided by the key exchange server) and an authentication tag, computed using key material extracted from

the NTS-KE handshake. The NTP server uses the cookie to recover this key material and send back an authenticated response. The response includes a fresh, encrypted cookie which the client then sends back in the clear in a subsequent request. (This constant refreshing of cookies is necessary in order to achieve NTS's unlinkability goal.)

Figure 1 provides an overview of the high-level interaction between the client, the NTS-KE server, and the NTP server. Note that the cookies' data format and the exchange of secrets between NTS-KE and NTP servers are not part of this specification and are implementation dependent. However, a suggested format for NTS cookies is provided in Section 6.

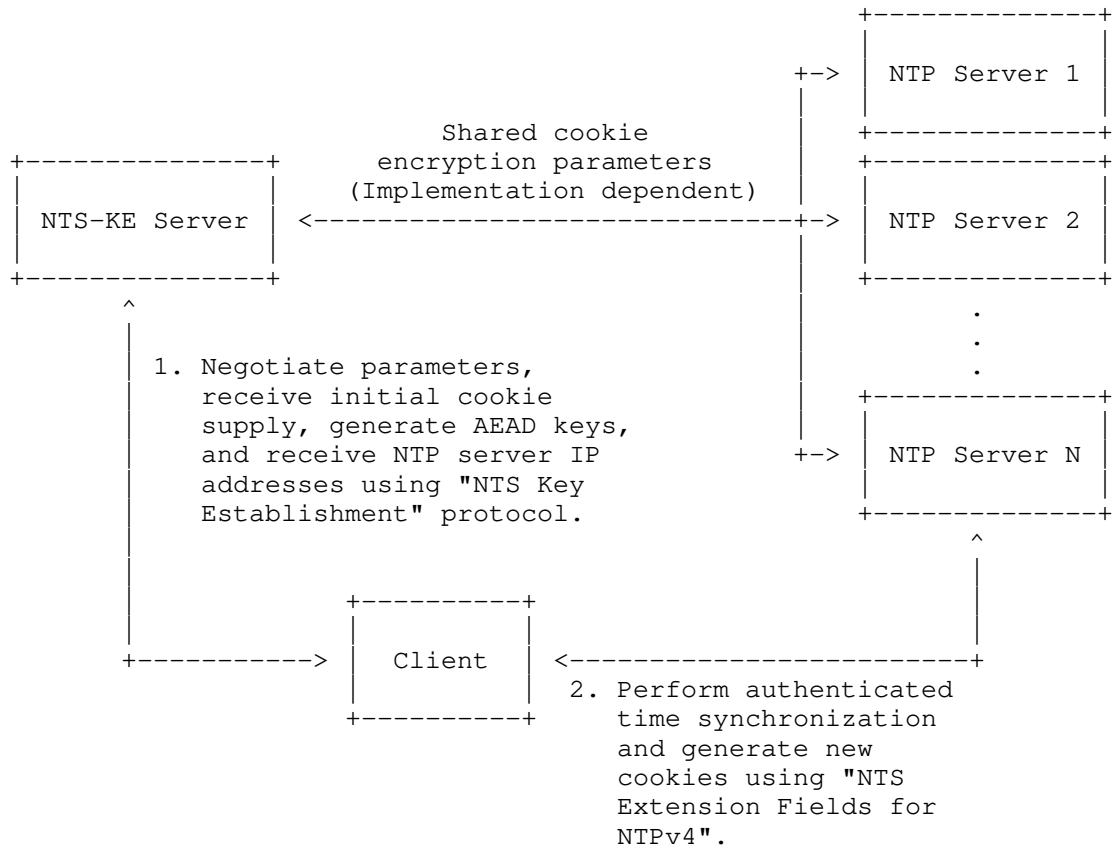


Figure 1: Overview of High-Level Interactions in NTS

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.

3. TLS profile for Network Time Security

Network Time Security makes use of TLS for NTS key establishment.

Since the NTS protocol is new as of this publication, no backward-compatibility concerns exist to justify using obsolete, insecure, or otherwise broken TLS features or versions. Implementations MUST conform with [RFC7525] or with a later revision of BCP 195. In particular, failure to use cipher suites that provide forward secrecy will make all negotiated NTS keys recoverable by anyone that gains access to the NTS-KE server's private certificate. Furthermore:

Implementations MUST NOT negotiate TLS versions earlier than 1.2, SHOULD negotiate TLS 1.3 [RFC8446] or later when possible, and MAY refuse to negotiate any TLS version which has been superseded by a later supported version.

Use of the Application-Layer Protocol Negotiation Extension [RFC7301] is integral to NTS and support for it is REQUIRED for interoperability.

4. The NTS Key Establishment Protocol

The NTS key establishment protocol is conducted via TCP port [[TBD1]]. The two endpoints carry out a TLS handshake in conformance with Section 3, with the client offering (via an ALPN [RFC7301] extension), and the server accepting, an application-layer protocol of "ntske/1". Immediately following a successful handshake, the client SHALL send a single request as Application Data encapsulated in the TLS-protected channel. Then, the server SHALL send a single response followed by a TLS "Close notify" alert and then discard the channel state.

The client's request and the server's response each SHALL consist of a sequence of records formatted according to Figure 2. Requests and non-error responses each SHALL include exactly one NTS Next Protocol Negotiation record. The sequence SHALL be terminated by a "End of Message" record. The requirement that all NTS-KE messages be terminated by an End of Message record makes them self-delimiting.

Clients and servers MAY enforce length limits on requests and responses, however, servers MUST accept requests of at least 1024 octets and clients SHOULD accept responses of at least 65536 octets.

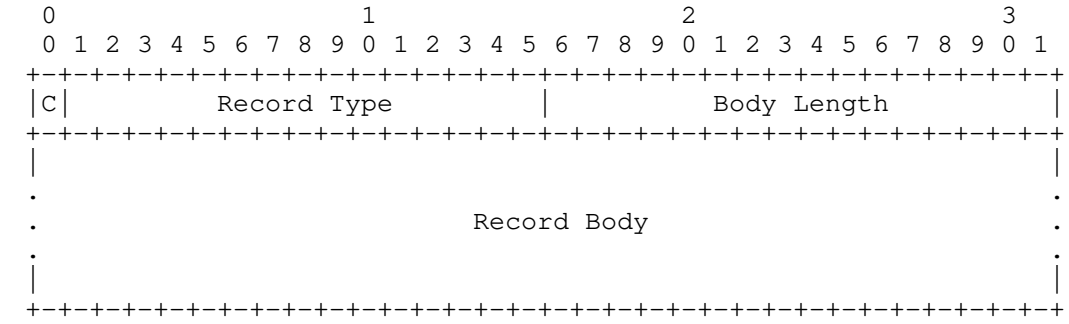


Figure 2: NTS-KE Record Format

The fields of an NTS-KE record are defined as follows:

C (Critical Bit): Determines the disposition of unrecognized Record Types. Implementations which receive a record with an unrecognized Record Type MUST ignore the record if the Critical Bit is 0 and MUST treat it as an error if the Critical Bit is 1.

Record Type Number: A 15-bit integer in network byte order. The semantics of record types 0-7 are specified in this memo. Additional type numbers SHALL be tracked through the IANA Network Time Security Key Establishment Record Types registry.

Body Length: The length of the Record Body field, in octets, as a 16-bit integer in network byte order. Record bodies MAY have any representable length and need not be aligned to a word boundary.

Record Body: The syntax and semantics of this field SHALL be determined by the Record Type.

For clarity regarding bit-endianness: the Critical Bit is the most-significant bit of the first octet. In C, given a network buffer `unsigned char b[]` containing an NTS-KE record, the critical bit is `b[0] >> 7` while the record type is `((b[0] & 0x7f) << 8) + b[1]`.

Figure 3 provides a schematic overview of the key exchange. It displays the protocol steps to be performed by the NTS client and server and record types to be exchanged.

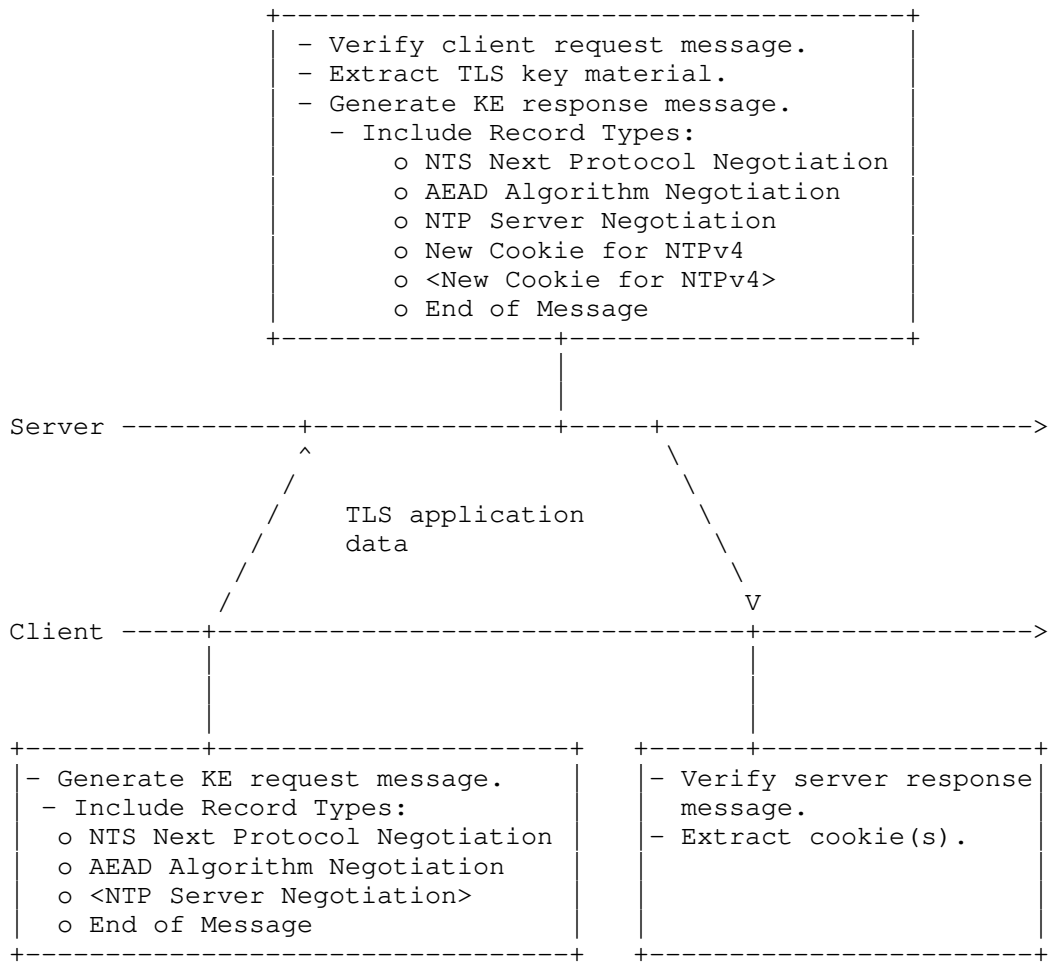


Figure 3: NTS Key Exchange Messages

4.1. NTS-KE Record Types

The following NTS-KE Record Types are defined:

4.1.1. End of Message

The End of Message record has a Record Type number of 0 and a zero-length body. It MUST occur exactly once as the final record of every NTS-KE request and response. The Critical Bit MUST be set.

4.1.2. NTS Next Protocol Negotiation

The NTS Next Protocol Negotiation record has a Record Type number of 1. It MUST occur exactly once in every NTS-KE request and response. Its body consists of a sequence of 16-bit unsigned integers in network byte order. Each integer represents a Protocol ID from the IANA Network Time Security Next Protocols registry. The Critical Bit MUST be set.

The Protocol IDs listed in the client's NTS Next Protocol Negotiation record denote those protocols which the client wishes to speak using the key material established through this NTS-KE session. The Protocol IDs listed in the server's response MUST comprise a subset of those listed in the request and denote those protocols which the server is willing and able to speak using the key material established through this NTS-KE session. The client MAY proceed with one or more of them. The request MUST list at least one protocol, but the response MAY be empty.

4.1.3. Error

The Error record has a Record Type number of 2. Its body is exactly two octets long, consisting of an unsigned 16-bit integer in network byte order, denoting an error code. The Critical Bit MUST be set.

Clients MUST NOT include Error records in their request. If clients receive a server response which includes an Error record, they MUST discard any negotiated key material and MUST NOT proceed to the Next Protocol.

The following error codes are defined:

Error code 0 means "Unrecognized Critical Record". The server MUST respond with this error code if the request included a record which the server did not understand and which had its Critical Bit set. The client SHOULD NOT retry its request without modification.

Error code 1 means "Bad Request". The server MUST respond with this error if, upon the expiration of an implementation-defined timeout, it has not yet received a complete and syntactically well-formed request from the client.

4.1.4. Warning

The Warning record has a Record Type number of 3. Its body is exactly two octets long, consisting of an unsigned 16-bit integer in

network byte order, denoting a warning code. The Critical Bit MUST be set.

Clients MUST NOT include Warning records in their request. If clients receive a server response which includes a Warning record, they MAY discard any negotiated key material and abort without proceeding to the Next Protocol. Unrecognized warning codes MUST be treated as errors.

This memo defines no warning codes.

4.1.5. AEAD Algorithm Negotiation

The AEAD Algorithm Negotiation record has a Record Type number of 4. Its body consists of a sequence of unsigned 16-bit integers in network byte order, denoting Numeric Identifiers from the IANA AEAD registry [RFC5116]. The Critical Bit MAY be set.

If the NTS Next Protocol Negotiation record offers Protocol ID 0 (for NTPv4), then this record MUST be included exactly once. Other protocols MAY require it as well.

When included in a request, this record denotes which AEAD algorithms the client is willing to use to secure the Next Protocol, in decreasing preference order. When included in a response, this record denotes which algorithm the server chooses to use. It is empty if the server supports none of the algorithms offered. In requests, the list MUST include at least one algorithm. In responses, it MUST include at most one. Honoring the client's preference order is OPTIONAL: servers may select among any of the client's offered choices, even if they are able to support some other algorithm which the client prefers more.

Server implementations of NTS extension fields for NTPv4 (Section 5) MUST support AEAD_AES_SIV_CMAC_256 [RFC5297] (Numeric Identifier 15). That is, if the client includes AEAD_AES_SIV_CMAC_256 in its AEAD Algorithm Negotiation record and the server accepts Protocol ID 0 (NTPv4) in its NTS Next Protocol Negotiation record, then the server's AEAD Algorithm Negotiation record MUST NOT be empty.

4.1.6. New Cookie for NTPv4

The New Cookie for NTPv4 record has a Record Type number of 5. The contents of its body SHALL be implementation-defined and clients MUST NOT attempt to interpret them. See Section 6 for a suggested construction.

Clients MUST NOT send records of this type. Servers MUST send at least one record of this type, and SHOULD send eight of them, if the Next Protocol Negotiation response record contains Protocol ID 0 (NTPv4) and the AEAD Algorithm Negotiation response record is not empty. The Critical Bit SHOULD NOT be set.

4.1.7. NTPv4 Server Negotiation

The NTPv4 Server Negotiation record has a Record Type number of 6. Its body consists of an ASCII-encoded [ANSI.X3-4.1986] string. The contents of the string SHALL be either an IPv4 address in dotted decimal notation, an IPv6 address, or a fully qualified domain name (FQDN). IPv6 addresses MUST conform to the "Text Representation of Addresses" as specified in [RFC4291] and MUST NOT include zone identifiers [RFC6874]. If internationalized labels are needed in the domain name, the A-LABEL syntax specified in [RFC5891] MUST be used.

When NTPv4 is negotiated as a Next Protocol and this record is sent by the server, the body specifies the hostname or IP address of the NTPv4 server with which the client should associate and which will accept the supplied cookies. If no record of this type is sent, the client SHALL interpret this as a directive to associate with an NTPv4 server at the same IP address as the NTS-KE server. Servers MUST NOT send more than one record of this type.

When this record is sent by the client, it indicates that the client wishes to associate with the specified NTP server. The NTS-KE server MAY incorporate this request when deciding what NTPv4 Server Negotiation records to respond with, but honoring the client's preference is OPTIONAL. The client MUST NOT send more than one record of this type.

Servers MAY set the Critical Bit on records of this type; clients SHOULD NOT.

4.1.8. NTPv4 Port Negotiation

The NTPv4 Port Negotiation record has a Record Type number of 7. Its body consists of a 16-bit unsigned integer in network byte order, denoting a UDP port number.

When NTPv4 is negotiated as a Next Protocol and this record is sent by the server, the body specifies the port number of the NTPv4 server with which the client should associate and which will accept the supplied cookies. If no record of this type is sent, the client SHALL assume a default of 123 (the registered port number for NTP).

When this record is sent by the client in conjunction with a NTPv4 Server Negotiation record, it indicates that the client wishes to associate with the NTP server at the specified port. The NTS-KE server MAY incorporate this request when deciding what NTPv4 Server Negotiation and NTPv4 Port Negotiation records to respond with, but honoring the client's preference is OPTIONAL.

Servers MAY set the Critical Bit on records of this type; clients SHOULD NOT.

4.2. Key Extraction (generally)

Following a successful run of the NTS-KE protocol, key material SHALL be extracted according to RFC 5705 [RFC5705]. Inputs to the exporter function are to be constructed in a manner specific to the negotiated Next Protocol. However, all protocols which utilize NTS-KE MUST conform to the following two rules:

The disambiguating label string MUST be "EXPORTER-network-time-security/1".

The per-association context value MUST be provided and MUST begin with the two-octet Protocol ID which was negotiated as a Next Protocol.

5. NTS Extension Fields for NTPv4

5.1. Key Extraction (for NTPv4)

Following a successful run of the NTS-KE protocol wherein Protocol ID 0 (NTPv4) is selected as a Next Protocol, two AEAD keys SHALL be extracted: a client-to-server (C2S) key and a server-to-client (S2C) key. These keys SHALL be computed according to RFC 5705 [RFC5705], using the following inputs.

The disambiguating label string SHALL be "EXPORTER-network-time-security/1".

The per-association context value SHALL consist of the following five octets:

The first two octets SHALL be zero (the Protocol ID for NTPv4).

The next two octets SHALL be the Numeric Identifier of the negotiated AEAD Algorithm in network byte order.

The final octet SHALL be 0x00 for the C2S key and 0x01 for the S2C key.

Implementations wishing to derive additional keys for private or experimental use MUST NOT do so by extending the above-specified syntax for per-association context values. Instead, they SHOULD use their own disambiguating label string. Note that RFC 5705 [RFC5705] provides that disambiguating label strings beginning with "EXPERIMENTAL" MAY be used without IANA registration.

5.2. Packet Structure Overview

In general, an NTS-protected NTPv4 packet consists of:

The usual 48-octet NTP header which is authenticated but not encrypted.

Some extension fields which are authenticated but not encrypted.

An extension field which contains AEAD output (i.e., an authentication tag and possible ciphertext). The corresponding plaintext, if non-empty, consists of some extension fields which benefit from both encryption and authentication.

Possibly, some additional extension fields which are neither encrypted nor authenticated. In general, these are discarded by the receiver.

Always included among the authenticated or authenticated-and-encrypted extension fields are a cookie extension field and a unique identifier extension field. The purpose of the cookie extension field is to enable the server to offload storage of session state onto the client. The purpose of the unique identifier extension field is to protect the client from replay attacks.

5.3. The Unique Identifier Extension Field

The Unique Identifier extension field provides the client with a cryptographically strong means of detecting replayed packets. It has a Field Type of [[TBD2]]. When the extension field is included in a client packet (mode 3), its body SHALL consist of a string of octets generated uniformly at random. The string MUST be at least 32 octets long. When the extension field is included in a server packet (mode 4), its body SHALL contain the same octet string as was provided in the client packet to which the server is responding. All server packets generated by NTS-implementing servers in response to client packets containing this extension field MUST also contain this field with the same content as in the client's request. The field's use in modes other than client-server is not defined.

This extension field MAY also be used standalone, without NTS, in which case it provides the client with a means of detecting spoofed packets from off-path attackers. Historically, NTP's origin timestamp field has played both these roles, but for cryptographic purposes this is suboptimal because it is only 64 bits long and, depending on implementation details, most of those bits may be predictable. In contrast, the Unique Identifier extension field enables a degree of unpredictability and collision resistance more consistent with cryptographic best practice.

5.4. The NTS Cookie Extension Field

The NTS Cookie extension field has a Field Type of [[TBD3]]. Its purpose is to carry information which enables the server to recompute keys and other session state without having to store any per-client state. The contents of its body SHALL be implementation-defined and clients MUST NOT attempt to interpret them. See Section 6 for a suggested construction. The NTS Cookie extension field MUST NOT be included in NTP packets whose mode is other than 3 (client) or 4 (server).

5.5. The NTS Cookie Placeholder Extension Field

The NTS Cookie Placeholder extension field has a Field Type of [[TBD4]]. When this extension field is included in a client packet (mode 3), it communicates to the server that the client wishes it to send additional cookies in its response. This extension field MUST NOT be included in NTP packets whose mode is other than 3.

Whenever an NTS Cookie Placeholder extension field is present, it MUST be accompanied by an NTS Cookie extension field. The body length of the NTS Cookie Placeholder extension field MUST be the same as the body length of the NTS Cookie extension field. This length requirement serves to ensure that the response will not be larger than the request, in order to improve timekeeping precision and prevent DDoS amplification. The contents of the NTS Cookie Placeholder extension field's body are undefined and, aside from checking its length, MUST be ignored by the server.

5.6. The NTS Authenticator and Encrypted Extension Fields Extension Field

The NTS Authenticator and Encrypted Extension Fields extension field is the central cryptographic element of an NTS-protected NTP packet. Its Field Type is [[TBD5]]. It SHALL be formatted according to Figure 4 and include the following fields:

Nonce Length: Two octets in network byte order, giving the length of the Nonce field, excluding any padding, interpreted as an unsigned integer.

Ciphertext Length: Two octets in network byte order, giving the length of the Ciphertext field, excluding any padding, interpreted as an unsigned integer.

Nonce: A nonce as required by the negotiated AEAD Algorithm. The field is zero-padded to a word (four octets) boundary.

Ciphertext: The output of the negotiated AEAD Algorithm. The structure of this field is determined by the negotiated algorithm, but it typically contains an authentication tag in addition to the actual ciphertext. The field is zero-padded to a word (four octets) boundary.

Additional Padding: Clients which use a nonce length shorter than the maximum allowed by the negotiated AEAD algorithm may be required to include additional zero-padding. The necessary length of this field is specified below.

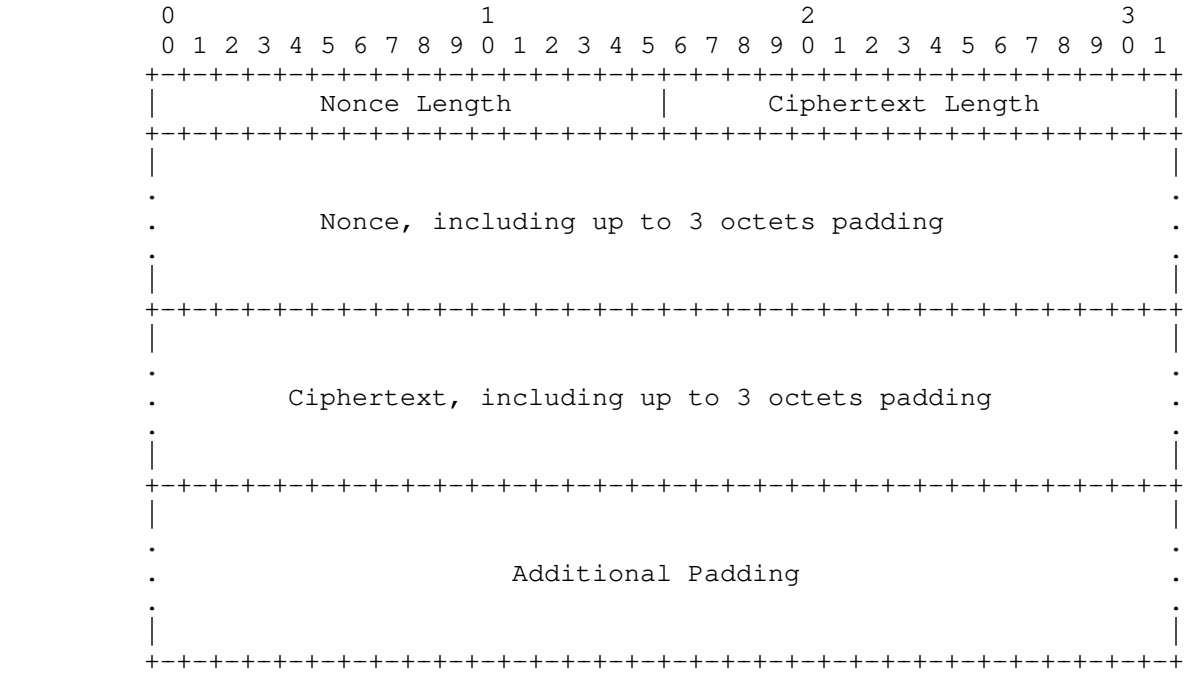


Figure 4: NTS Authenticator and Encrypted Extension Fields Extension Field Format

The Ciphertext field SHALL be formed by providing the following inputs to the negotiated AEAD Algorithm:

K: For packets sent from the client to the server, the C2S key SHALL be used. For packets sent from the server to the client, the S2C key SHALL be used.

A: The associated data SHALL consist of the portion of the NTP packet beginning from the start of the NTP header and ending at the end of the last extension field which precedes the NTS Authenticator and Encrypted Extension Fields extension field.

P: The plaintext SHALL consist of all (if any) NTP extension fields to be encrypted. The format of any such fields SHALL be in accordance with RFC 7822 [RFC7822]. If multiple extension fields are present they SHALL be joined by concatenation.

N: The nonce SHALL be formed however required by the negotiated AEAD algorithm.

The purpose of the Additional Padding field is to ensure that servers can always choose a nonce whose length is adequate to ensure its uniqueness, even if the client chooses a shorter one, and still ensure that the overall length of the server's response packet does not exceed the length of the request. For mode 4 (server) packets, no Additional Padding field is ever required. For mode 3 (client) packets, the length of the Additional Padding field SHALL be computed as follows. Let 'N_LEN' be the padded length of the the Nonce field. Let 'N_MAX' be, as specified by RFC 5116 [RFC5116], the maximum permitted nonce length for the negotiated AEAD algorithm. Let 'N_REQ' be the lesser of 16 and N_MAX, rounded up to the nearest multiple of 4. If N_LEN is greater than or equal to N_REQ, then no Additional Padding field is required. Otherwise, the Additional Padding field SHALL be at least N_REQ - N_LEN octets in length. Servers MUST enforce this requirement by discarding any packet which does not conform to it.

The NTS Authenticator and Encrypted Extension Fields extension field MUST NOT be included in NTP packets whose mode is other than 3 (client) or 4 (server).

5.7. Protocol Details

A client sending an NTS-protected request SHALL include the following extension fields as displayed in Figure 5:

Exactly one Unique Identifier extension field which MUST be authenticated, MUST NOT be encrypted, and whose contents MUST NOT duplicate those of any previous request.

Exactly one NTS Cookie extension field which MUST be authenticated and MUST NOT be encrypted. The cookie MUST be one which has been previously provided to the client; either from the key exchange server during the NTS-KE handshake or from the NTP server in response to a previous NTS-protected NTP request.

Exactly one NTS Authenticator and Encrypted Extension Fields extension field, generated using an AEAD Algorithm and C2S key established through NTS-KE.

To protect the client's privacy, the client SHOULD avoid reusing a cookie. If the client does not have any cookies that it has not already sent, it SHOULD initiate a re-run the NTS-KE protocol. The client MAY reuse cookies in order to prioritize resilience over unlinkability. Which of the two that should be prioritized in any particular case is dependent on the application and the user's preference. Section 10.1 describes the privacy considerations of this in further detail.

The client MAY include one or more NTS Cookie Placeholder extension fields which MUST be authenticated and MAY be encrypted. The number of NTS Cookie Placeholder extension fields that the client includes SHOULD be such that if the client includes N placeholders and the server sends back N+1 cookies, the number of unused cookies stored by the client will come to eight. The client SHOULD NOT include more than seven NTS Cookie Placeholder extension fields in a request. When both the client and server adhere to all cookie-management guidance provided in this memo, the number of placeholder extension fields will equal the number of dropped packets since the last successful volley.

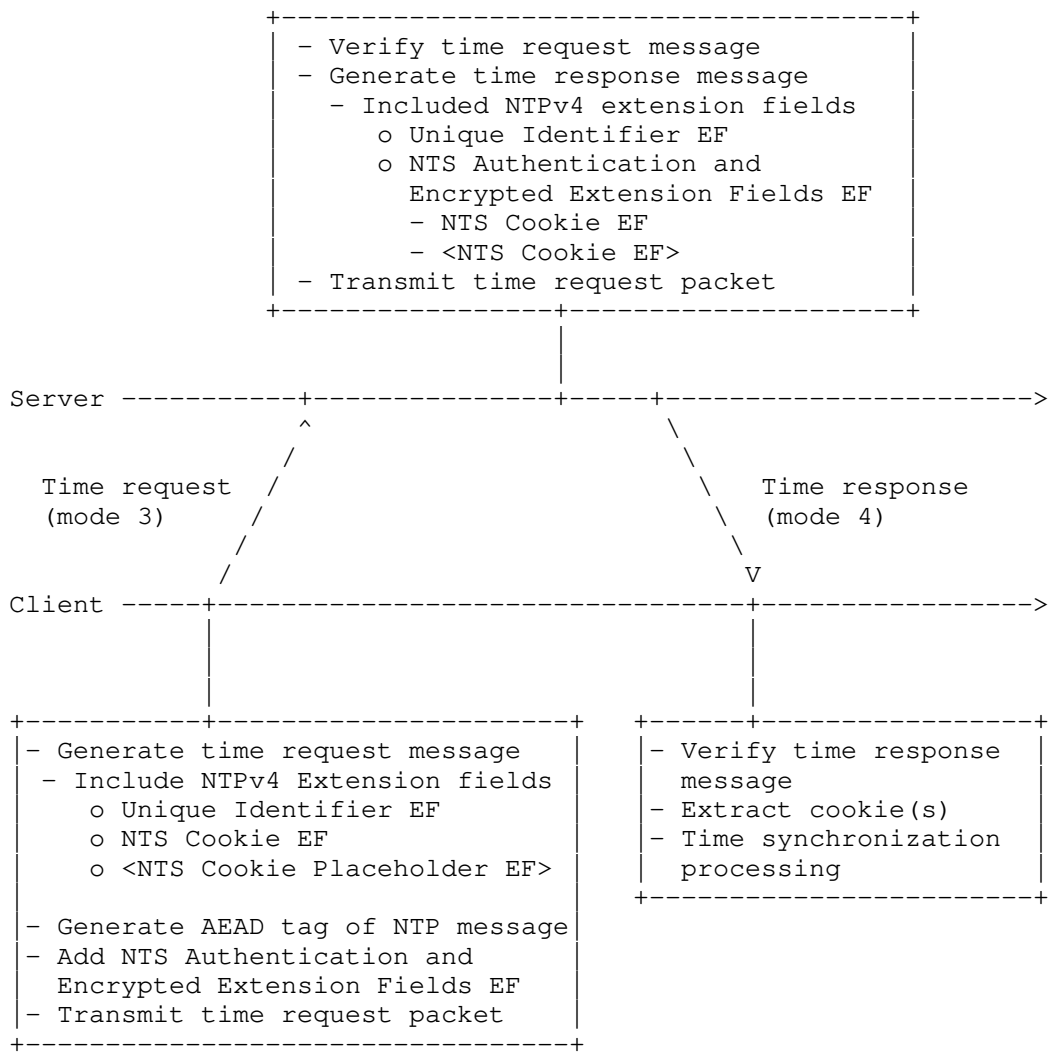


Figure 5: NTS Time Synchronization Messages

The client MAY include additional (non-NTS-related) extension fields which MAY appear prior to the NTS Authenticator and Encrypted Extension Fields extension fields (therefore authenticated but not encrypted), within it (therefore encrypted and authenticated), or after it (therefore neither encrypted nor authenticated). In general, however, the server MUST discard any unauthenticated extension fields and process the packet as though they were not present. Servers MAY implement exceptions to this requirement for

particular extension fields if their specification explicitly provides for such.

Upon receiving an NTS-protected request, the server SHALL (through some implementation-defined mechanism) use the cookie to recover the AEAD Algorithm, C2S key, and S2C key associated with the request, and then use the C2S key to authenticate the packet and decrypt the ciphertext. If the cookie is valid and authentication and decryption succeed, the server SHALL include the following extension fields in its response:

- Exactly one Unique Identifier extension field which MUST be authenticated, MUST NOT be encrypted, and whose contents SHALL echo those provided by the client.

- Exactly one NTS Authenticator and Encrypted Extension Fields extension field, generated using the AEAD algorithm and S2C key recovered from the cookie provided by the client.

- One or more NTS Cookie extension fields which MUST be authenticated and encrypted. The number of NTS Cookie extension fields included SHOULD be equal to, and MUST NOT exceed, one plus the number of valid NTS Cookie Placeholder extension fields included in the request. The cookies returned in those fields MUST be valid for use with the NTP server that sent them. They MAY be valid for other NTP servers as well, but there is no way for the server to indicate this.

We emphasize the contrast that NTS Cookie extension fields MUST NOT be encrypted when sent from client to server, but MUST be encrypted when sent from server to client. The former is necessary in order for the server to be able to recover the C2S and S2C keys, while the latter is necessary to satisfy the unlinkability goals discussed in Section 10.1. We emphasize also that "encrypted" means encapsulated within the the NTS Authenticator and Encrypted Extensions extension field. While the body of an NTS Cookie extension field will generally consist of some sort of AEAD output (regardless of whether the recommendations of Section 6 are precisely followed), this is not sufficient to make the extension field "encrypted".

The server MAY include additional (non-NTS-related) extension fields which MAY appear prior to the NTS Authenticator and Encrypted Extension Fields extension field (therefore authenticated but not encrypted), within it (therefore encrypted and authenticated), or after it (therefore neither encrypted nor authenticated). In general, however, the client MUST discard any unauthenticated extension fields and process the packet as though they were not present. Clients MAY implement exceptions to this requirement for

particular extension fields if their specification explicitly provides for such.

Upon receiving an NTS-protected response, the client MUST verify that the Unique Identifier matches that of an outstanding request, and that the packet is authentic under the S2C key associated with that request. If either of these checks fails, the packet MUST be discarded without further processing.

If the server is unable to validate the cookie or authenticate the request, it SHOULD respond with a Kiss-o'-Death (KoD) packet (see RFC 5905, Section 7.4 [RFC5905]) with kiss code "NTSN", meaning "NTS negative-acknowledgment (NAK)". It MUST NOT include any NTS Cookie or NTS Authenticator and Encrypted Extension Fields extension fields.

If the NTP server has previously responded with authentic NTS-protected NTP packets (i.e., packets containing the NTS Authenticator and Encrypted Extension Fields extension field), the client MUST verify that any KoD packets received from the server contain the Unique Identifier extension field and that the Unique Identifier matches that of an outstanding request. If this check fails, the packet MUST be discarded without further processing. If this check passes, the client MUST comply with RFC 5905, Section 7.4 [RFC5905] where required. A client MAY automatically re-run the NTS-KE protocol upon forced disassociation from an NTP server. In that case, it MUST be able to detect and stop looping between the NTS-KE and NTP servers by rate limiting the retries using e.g. exponential retry intervals.

Upon reception of the NTS NAK kiss code, the client SHOULD wait until the next poll for a valid NTS-protected response and if none is received, initiate a fresh NTS-KE handshake to try to renegotiate new cookies, AEAD keys, and parameters. If the NTS-KE handshake succeeds, the client MUST discard all old cookies and parameters and use the new ones instead. As long as the NTS-KE handshake has not succeeded, the client SHOULD continue polling the NTP server using the cookies and parameters it has.

To allow for NTP session restart when the NTS-KE server is unavailable and to reduce NTS-KE server load, the client SHOULD keep at least one unused but recent cookie, AEAD keys, negotiated AEAD algorithm, and other necessary parameters on persistent storage. This way, the client is able to resume the NTP session without performing renewed NTS-KE negotiation.

6. Suggested Format for NTS Cookies

This section is non-normative. It gives a suggested way for servers to construct NTS cookies. All normative requirements are stated in Section 4.1.6 and Section 5.4.

The role of cookies in NTS is closely analogous to that of session cookies in TLS. Accordingly, the thematic resemblance of this section to RFC 5077 [RFC5077] is deliberate and the reader should likewise take heed of its security considerations.

Servers should select an AEAD algorithm which they will use to encrypt and authenticate cookies. The chosen algorithm should be one such as AEAD_AES_SIV_CMAC_256 [RFC5297] which resists accidental nonce reuse. It need not be the same as the one that was negotiated with the client. Servers should randomly generate and store a master AEAD key 'K'. Servers should additionally choose a non-secret, unique value 'I' as key-identifier for 'K'.

Servers should periodically (e.g., once daily) generate a new pair (I,K) and immediately switch to using these values for all newly-generated cookies. Immediately following each such key rotation, servers should securely erase any keys generated two or more rotation periods prior. Servers should continue to accept any cookie generated using keys that they have not yet erased, even if those keys are no longer current. Erasing old keys provides for forward secrecy, limiting the scope of what old information can be stolen if a master key is somehow compromised. Holding on to a limited number of old keys allows clients to seamlessly transition from one generation to the next without having to perform a new NTS-KE handshake.

The need to keep keys synchronized between NTS-KE and NTP servers as well as across load-balanced clusters can make automatic key rotation challenging. However, the task can be accomplished without the need for central key-management infrastructure by using a ratchet, i.e., making each new key a deterministic, cryptographically pseudo-random function of its predecessor. A recommended concrete implementation of this approach is to use HKDF [RFC5869] to derive new keys, using the key's predecessor as Input Keying Material and its key identifier as a salt.

To form a cookie, servers should first form a plaintext 'P' consisting of the following fields:

The AEAD algorithm negotiated during NTS-KE.

The S2C key.

The C2S key.

Servers should then generate a nonce `N` uniformly at random, and form AEAD output `C` by encrypting `P` under key `K` with nonce `N` and no associated data.

The cookie should consist of the tuple `(I,N,C)`.

To verify and decrypt a cookie provided by the client, first parse it into its components `I`, `N`, and `C`. Use `I` to look up its decryption key `K`. If the key whose identifier is `I` has been erased or never existed, decryption fails; reply with an NTS NAK. Otherwise, attempt to decrypt and verify ciphertext `C` using key `K` and nonce `N` with no associated data. If decryption or verification fails, reply with an NTS NAK. Otherwise, parse out the contents of the resulting plaintext `P` to obtain the negotiated AEAD algorithm, S2C key, and C2S key.

7. IANA Considerations

7.1. Service Name and Transport Protocol Port Number Registry

IANA is requested to allocate the following entry in the Service Name and Transport Protocol Port Number Registry [RFC6335]:

Service Name: ntske

Transport Protocol: tcp

Assignee: IESG <iesg@ietf.org>

Contact: IETF Chair <chair@ietf.org>

Description: Network Time Security Key Exchange

Reference: [[this memo]]

Port Number: [[TBD1]], selected by IANA from the User Port range

[[RFC EDITOR: Replace all instances of [[TBD1]] in this document with the IANA port assignment.]]

7.2. TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs Registry

IANA is requested to allocate the following entry in the TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs registry [RFC7301]:

Protocol: Network Time Security Key Establishment, version 1

Identification Sequence:
0x6E 0x74 0x73 0x6B 0x65 0x2F 0x31 ("ntske/1")

Reference: [[this memo]], Section 4

7.3. TLS Exporter Labels Registry

IANA is requested to allocate the following entry in the TLS Exporter Labels Registry [RFC5705]:

Value	DTLS-OK	Recommended	Reference	Note
EXPORTER-network-time-security/1	Y	Y	[[this memo]], Section 4.2	

7.4. NTP Kiss-o'-Death Codes Registry

IANA is requested to allocate the following entry in the registry of NTP Kiss-o'-Death Codes [RFC5905]:

Code	Meaning	Reference
NTSN	Network Time Security (NTS) negative-acknowledgment (NAK)	[[this memo]], Section 5.7

7.5. NTP Extension Field Types Registry

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

Field Type	Meaning	Reference
[[TBD2]]	Unique Identifier	[[this memo]], Section 5.3
[[TBD3]]	NTS Cookie	[[this memo]], Section 5.4
[[TBD4]]	NTS Cookie Placeholder	[[this memo]], Section 5.5
[[TBD5]]	NTS Authenticator and Encrypted Extension Fields	[[this memo]], Section 5.6

[[RFC EDITOR: Replace all instances of [[TBD2]], [[TBD3]], [[TBD4]], and [[TBD5]] in this document with the respective IANA assignments.

7.6. Network Time Security Key Establishment Record Types Registry

IANA is requested to create a new registry entitled "Network Time Security Key Establishment Record Types". Entries SHALL have the following fields:

Record Type Number (REQUIRED): An integer in the range 0-32767 inclusive.

Description (REQUIRED): A short text description of the purpose of the field.

Reference (REQUIRED): A reference to a document specifying the semantics of the record.

The policy for allocation of new entries in this registry SHALL vary by the Record Type Number, as follows:

0-1023: IETF Review

1024-16383: Specification Required

16384-32767: Private and Experimental Use

Applications for new entries SHALL specify the contents of the Description, Set Critical Bit, and Reference fields as well as which of the above ranges the Record Type Number should be allocated from. Applicants MAY request a specific Record Type Number and such requests MAY be granted at the registrar’s discretion.

The initial contents of this registry SHALL be as follows:

Record Type Number	Description	Reference
0	End of Message	[[this memo]], Section 4.1.1
1	NTS Next Protocol Negotiation	[[this memo]], Section 4.1.2
2	Error	[[this memo]], Section 4.1.3
3	Warning	[[this memo]], Section 4.1.4
4	AEAD Algorithm Negotiation	[[this memo]], Section 4.1.5
5	New Cookie for NTPv4	[[this memo]], Section 4.1.6
6	NTPv4 Server Negotiation	[[this memo]], Section 4.1.7
7	NTPv4 Port Negotiation	[[this memo]], Section 4.1.8
16384-32767	Reserved for Private & Experimental Use	[[this memo]]

7.7. Network Time Security Next Protocols Registry

IANA is requested to create a new registry entitled "Network Time Security Next Protocols". Entries SHALL have the following fields:

Protocol ID (REQUIRED): An integer in the range 0-65535 inclusive, functioning as an identifier.

Protocol Name (REQUIRED): A short text string naming the protocol being identified.

Reference (REQUIRED): A reference to a relevant specification document.

The policy for allocation of new entries in these registries SHALL vary by their Protocol ID, as follows:

0-1023: IETF Review

1024-32767: Specification Required

32768-65535: Private and Experimental Use

The initial contents of this registry SHALL be as follows:

Protocol ID	Protocol Name	Reference
0	Network Time Protocol version 4 (NTPv4)	[[this memo]]
32768-65535	Reserved for Private or Experimental Use	Reserved by [[this memo]]

7.8. Network Time Security Error and Warning Codes Registries

IANA is requested to create two new registries entitled "Network Time Security Error Codes" and "Network Time Security Warning Codes". Entries in each SHALL have the following fields:

Number (REQUIRED): An integer in the range 0-65535 inclusive

Description (REQUIRED): A short text description of the condition.

Reference (REQUIRED): A reference to a relevant specification document.

The policy for allocation of new entries in these registries SHALL vary by their Number, as follows:

0-1023: IETF Review

1024-32767: Specification Required

32768-65535: Private and Experimental Use

The initial contents of the Network Time Security Error Codes Registry SHALL be as follows:

Number	Description	Reference
0	Unrecognized Critical Extension	[[this memo]], Section 4.1.3
1	Bad Request	[[this memo]], Section 4.1.3
32768-65535	Reserved for Private or Experimental Use	Reserved by [[this memo]]

The Network Time Security Warning Codes Registry SHALL initially be empty except for the reserved range, i.e.:

Number	Description	Reference
32768-65535	Reserved for Private or Experimental Use	Reserved by [[this memo]]

8. 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".

8.1. Implementation PoC 1

Organization: Ostfalia University of Applied Science

Implementor: Martin Langer

Maturity: Proof-of-Concept Prototype

This implementation was used to verify consistency and to ensure completeness of this specification. It also demonstrate interoperability with NTP's client-server mode messages.

8.1.1. Coverage

This implementation covers the complete specification.

8.1.2. Licensing

The code is released under a Apache License 2.0 license.

The source code is available at: https://gitlab.com/MLanger/nts/

8.1.3. Contact Information

Contact Martin Langer: mart.langer@ostfalia.de

8.1.4. Last Update

The implementation was updated 3rd May 2018.

8.2. Implementation PoC 2

Organization: Akamai Technologies

Implementor: Daniel Fox Franke

Maturity: Proof-of-Concept Prototype

This implementation was used to verify consistency and to ensure completeness of this specification.

8.2.1. Coverage

This implementation provides the client and the server for the initial TLS handshake and NTS key exchange. It provides the the client part of the NTS protected NTP messages.

8.2.2. Licensing

Public domain.

The source code is available at: https://github.com/dfoxfranke/nts-hackathon

8.2.3. Contact Information

Contact Daniel Fox Franke: dfoxfranke@gmail.com

8.2.4. Last Update

The implementation was updated 16th March 2018.

8.3. Interoperability

The Interoperability tests distinguished between NTS key exchange and NTS time exchange messages. For the NTS key exchange, interoperability between the two implementations has been verified successfully. Interoperability of NTS time exchange messages has been verified successfully for the case that PoC 1 represents the server and PoC 2 the client.

These tests successfully demonstrate that there are at least two running implementations of this draft which are able to interoperate.

9. Security Considerations

9.1. Sensitivity to DDoS attacks

The introduction of NTS brings with it the introduction of asymmetric cryptography to NTP. Asymmetric cryptography is necessary for initial server authentication and AEAD key extraction. Asymmetric cryptosystems are generally orders of magnitude slower than their symmetric counterparts. This makes it much harder to build systems that can serve requests at a rate corresponding to the full line speed of the network connection. This, in turn, opens up a new possibility for DDoS attacks on NTP services.

The main protection against these attacks in NTS lies in that the use of asymmetric cryptosystems is only necessary in the initial NTS-KE phase of the protocol. Since the protocol design enables separation of the NTS-KE and NTP servers, a successful DDoS attack on an NTS-KE server separated from the NTP service it supports will not affect NTP users that have already performed initial authentication, AEAD key extraction, and cookie exchange.

NTS users should also consider that they are not fully protected against DDoS attacks by on-path adversaries. In addition to dropping packets and attacks such as those described in Section 9.4, an on-path attacker can send spoofed kiss-o'-death replies, which are not authenticated, in response to NTP requests. This could result in significantly increased load on the NTS-KE server. Implementers have to weigh the user's need for unlinkability against the added resilience that comes with cookie reuse in cases of NTS-KE server unavailability.

9.2. Avoiding DDoS Amplification

Certain non-standard and/or deprecated features of the Network Time Protocol enable clients to send a request to a server which causes the server to send a response much larger than the request. Servers

which enable these features can be abused in order to amplify traffic volume in DDoS attacks by sending them a request with a spoofed source IP. In recent years, attacks of this nature have become an endemic nuisance.

NTS is designed to avoid contributing any further to this problem by ensuring that NTS-related extension fields included in server responses will be the same size as the NTS-related extension fields sent by the client. In particular, this is why the client is required to send a separate and appropriately padded-out NTS Cookie Placeholder extension field for every cookie it wants to get back, rather than being permitted simply to specify a desired quantity.

Due to the RFC 7822 [RFC7822] requirement that extensions be padded and aligned to four-octet boundaries, response size may still in some cases exceed request size by up to three octets. This is sufficiently inconsequential that we have declined to address it.

9.3. Initial Verification of Server Certificates

NTS's security goals are undermined if the client fails to verify that the X.509 certificate chain presented by the NTS-KE server is valid and rooted in a trusted certificate authority. RFC 5280 [RFC5280] and RFC 6125 [RFC6125] specify how such verification is to be performed in general. However, the expectation that the client does not yet have a correctly-set system clock at the time of certificate verification presents difficulties with verifying that the certificate is within its validity period, i.e., that the current time lies between the times specified in the certificate's notBefore and notAfter fields. It may be operationally necessary in some cases for a client to accept a certificate which appears to be expired or not yet valid. While there is no perfect solution to this problem, there are several mitigations the client can implement to make it more difficult for an adversary to successfully present an expired certificate:

Check whether the system time is in fact unreliable. If the system clock has previously been synchronized since last boot, then on operating systems which implement a kernel-based phase-locked-loop API, a call to ntp_gettime() should show a maximum error less than NTP_PHASE_MAX. In this case, the clock SHOULD be considered reliable and certificates can be strictly validated.

Allow the system administrator to specify that certificates should *always* be strictly validated. Such a configuration is appropriate on systems which have a battery-backed clock and which can reasonably prompt the user to manually set an approximately-correct time if it appears to be needed.

Once the clock has been synchronized, periodically write the current system time to persistent storage. Do not accept any certificate whose notAfter field is earlier than the last recorded time.

NTP time replies are expected to be consistent with the NTS-KE TLS certificate validity period, i.e. time replies received immediately after an NTS-KE handshake are expected to lie within the certificate validity period. Implementations are recommended to check that this is the case. Performing a new NTS-KE handshake based solely on the fact that the certificate used by the NTS-KE server in a previous handshake has expired is normally not necessary. Clients that still wish to do this must take care not to cause an inadvertent denial-of-service attack on the NTS-KE server, for example by picking a random time in the week preceding certificate expiry to perform the new handshake.

Use multiple time sources. The ability to pass off an expired certificate is only useful to an adversary who has compromised the corresponding private key. If the adversary has compromised only a minority of servers, NTP's selection algorithm (RFC 5905 section 11.2.1 [RFC5905]) will protect the client from accepting bad time from the adversary-controlled servers.

9.4. Delay Attacks

In a packet delay attack, an adversary with the ability to act as a man-in-the-middle delays time synchronization packets between client and server asymmetrically [RFC7384]. Since NTP's formula for computing time offset relies on the assumption that network latency is roughly symmetrical, this leads to the client to compute an inaccurate value [Mizrahi]. The delay attack does not reorder or modify the content of the exchanged synchronization packets. Therefore, cryptographic means do not provide a feasible way to mitigate this attack. However, the maximum error that an adversary can introduce is bounded by half of the round trip delay.

RFC 5905 [RFC5905] specifies a parameter called MAXDIST which denotes the maximum round-trip latency (including not only the immediate round trip between client and server, but the whole distance back to the reference clock as reported in the Root Delay field) that a client will tolerate before concluding that the server is unsuitable for synchronization. The standard value for MAXDIST is one second, although some implementations use larger values. Whatever value a client chooses, the maximum error which can be introduced by a delay attack is MAXDIST/2.

Usage of multiple time sources, or multiple network paths to a given time source [Shpiner], may also serve to mitigate delay attacks if the adversary is in control of only some of the paths.

9.5. Random Number Generation

At various points in NTS, the generation of cryptographically secure random numbers is required. Whenever this draft specifies the use of random numbers, cryptographically secure random number generation MUST be used. RFC 4086 [RFC4086] contains guidelines concerning this topic.

9.6. NTS Stripping

Implementers must be aware of the possibility of "NTS stripping" attacks, where an attacker tricks clients into reverting to plain NTP. Naive client implementations might, for example, revert automatically to plain NTP if the NTS-KE handshake fails. A man-in-the-middle attacker can easily cause this to happen. Even clients that already hold valid cookies can be vulnerable, since an attacker can force a client to repeat the NTS-KE handshake by sending faked NTP mode 4 replies with the NTS NAK kiss code. Forcing a client to repeat the NTS-KE handshake can also be the first step in more advanced attacks.

For the reasons described here, implementations SHOULD NOT revert from NTS-protected to unprotected NTP with any server without explicit user action.

10. Privacy Considerations

10.1. Unlinkability

Unlinkability prevents a device from being tracked when it changes network addresses (e.g. because said device moved between different networks). In other words, unlinkability thwarts an attacker that seeks to link a new network address used by a device with a network address that it was formerly using, because of recognizable data that the device persistently sends as part of an NTS-secured NTP association. This is the justification for continually supplying the client with fresh cookies, so that a cookie never represents recognizable data in the sense outlined above.

NTS's unlinkability objective is merely to not leak any additional data that could be used to link a device's network address. NTS does not rectify legacy linkability issues that are already present in NTP. Thus, a client that requires unlinkability must also minimize

information transmitted in a client query (mode 3) packet as described in the draft [I-D.ietf-ntp-data-minimization].

The unlinkability objective only holds for time synchronization traffic, as opposed to key exchange traffic. This implies that it cannot be guaranteed for devices that function not only as time clients, but also as time servers (because the latter can be externally triggered to send authentication data).

It should also be noted that it could be possible to link devices that operate as time servers from their time synchronization traffic, using information exposed in (mode 4) server response packets (e.g. reference ID, reference time, stratum, poll). Also, devices that respond to NTP control queries could be linked using the information revealed by control queries.

Note that the unlinkability objective does not prevent a client device to be tracked by its time servers.

10.2. Confidentiality

NTS does not protect the confidentiality of information in NTP's header fields. When clients implement [I-D.ietf-ntp-data-minimization], client packet headers do not contain any information which the client could conceivably wish to keep secret: one field is random, and all others are fixed. Information in server packet headers is likewise public: the origin timestamp is copied from the client's (random) transmit timestamp, and all other fields are set the same regardless of the identity of the client making the request.

Future extension fields could hypothetically contain sensitive information, in which case NTS provides a mechanism for encrypting them.

11. Acknowledgements

The authors would like to thank Richard Barnes, Steven Bellovin, Patrik Faeltstroem (Faltstrom), Scott Fluhrer, Sharon Goldberg, Russ Housley, Martin Langer, Miroslav Lichvar, Aanchal Malhotra, Dave Mills, Danny Mayer, Karen O'Donoghue, Eric K. Rescorla, Stephen Roettger, Kurt Roeckx, Kyle Rose, Rich Salz, Brian Sniffen, Susan Sons, Douglas Stebila, Harlan Stenn, Joachim Stroembergsson (Strombergsson), Martin Thomson, Richard Welty, and Christer Weinigel for contributions to this document and comments on the design of NTS.

12. References

12.1. Normative References

[ANSI.X3-4.1986]
American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[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>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, DOI 10.17487/RFC5116, January 2008, <https://www.rfc-editor.org/info/rfc5116>.

[RFC5297] Harkins, D., "Synthetic Initialization Vector (SIV) Authenticated Encryption Using the Advanced Encryption Standard (AES)", RFC 5297, DOI 10.17487/RFC5297, October 2008, <https://www.rfc-editor.org/info/rfc5297>.

[RFC5705] Rescorla, E., "Keying Material Exporters for Transport Layer Security (TLS)", RFC 5705, DOI 10.17487/RFC5705, March 2010, <https://www.rfc-editor.org/info/rfc5705>.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <https://www.rfc-editor.org/info/rfc5891>.

[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>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC6874] Carpenter, B., Cheshire, S., and R. Hinden, "Representing IPv6 Zone Identifiers in Address Literals and Uniform Resource Identifiers", RFC 6874, DOI 10.17487/RFC6874, February 2013, <https://www.rfc-editor.org/info/rfc6874>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

[RFC7507] Moeller, B. and A. Langley, "TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks", RFC 7507, DOI 10.17487/RFC7507, April 2015, <https://www.rfc-editor.org/info/rfc7507>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[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>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

12.2. Informative References

[I-D.ietf-ntp-data-minimization]
Franke, D. and A. Malhotra, "NTP Client Data Minimization", draft-ietf-ntp-data-minimization-02 (work in progress), July 2018.

[Mizrahi] Mizrahi, T., "A game theoretic analysis of delay attacks against time synchronization protocols", in Proceedings of Precision Clock Synchronization for Measurement Control and Communication, ISPCS 2012, pp. 1-6, September 2012.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, DOI 10.17487/RFC5077, January 2008, <https://www.rfc-editor.org/info/rfc5077>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/info/rfc5869>.

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <https://www.rfc-editor.org/info/rfc7384>.

[Shpiner] "Multi-path Time Protocols", in Proceedings of IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), September 2013.

Appendix A. Terms and Abbreviations

AEAD Authenticated Encryption with Associated Data [RFC5116]

ALPN Application-Layer Protocol Negotiation [RFC7301]

C2S Client-to-server

DDoS Distributed Denial-of-Service

EF Extension Field [RFC5905]

HKDF Hashed Message Authentication Code-based Key Derivation Function [RFC5869]

IANA Internet Assigned Numbers Authority

IP Internet Protocol

KoD Kiss-o'-Death [RFC5905]

NTP Network Time Protocol [RFC5905]

NTS Network Time Security

NTS-KE Network Time Security Key Exchange

S2C Server-to-client

SCSV Signaling Cipher Suite Value [RFC7507]

TCP Transmission Control Protocol [RFC0793]

TLS Transport Layer Security [RFC8446]

UDP User Datagram Protocol [RFC0768]

Authors' Addresses

Daniel Fox Franke
Akamai Technologies
150 Broadway
Cambridge, MA 02142
United States

Email: dafranke@akamai.com
URI: https://www.dfranke.us

Dieter Sibold
Physikalisch-Technische Bundesanstalt
Bundesallee 100
Braunschweig D-38116
Germany

Phone: +49-(0)531-592-8420
Fax: +49-531-592-698420
Email: dieter.sibold@ptb.de

Kristof Teichel
Physikalisch-Technische Bundesanstalt
Bundesallee 100
Braunschweig D-38116
Germany

Phone: +49-(0)531-592-4471
Email: kristof.teichel@ptb.de

Marcus Dansarie

Email: marcus@dansarie.se

Ragnar Sundblad
Netnod

Email: ragge@netnod.se

Internet Working Group
Y. Jiang, Ed.
Huawei
Internet-Draft
X. Liu
Independent
Intended status: Standards Track
J. Xu
Huawei
R. Cummings, Ed.
National Instruments
Expires: July 2019
January 3, 2019

YANG Data Model for IEEE 1588-2008
draft-ietf-tictoc-1588v2-yang-11

Abstract

This document defines a YANG data model for the configuration of IEEE 1588-2008 devices and clocks, and also retrieval of the configuration information, data set and running states of IEEE 1588-2008 clocks. The YANG module in this document conforms to the Network Management Datastore Architecture (NMDA).

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on July 3, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Conventions used in this document 4
1.2. Terminology 4
2. IEEE 1588-2008 YANG Model hierarchy 5
2.1. Interpretations from IEEE 1588 Working Group 8
2.2. Configuration and state 8
3. IEEE 1588-2008 YANG Module 9
4. Security Considerations 22
5. IANA Considerations 23
6. References 23
6.1. Normative References 23
6.2. Informative References 24
7. Acknowledgments 25
Appendix A Transferring YANG Work to IEEE 1588 WG 26
A.1. Assumptions for the Transfer 27
A.2. Intellectual Property Considerations 27
A.3. Namespace and Module Name 28
A.4. IEEE 1588 YANG Modules in ASCII Format 29

1. Introduction

As a synchronization protocol, IEEE 1588-2008 [IEEE1588] is widely supported in the carrier networks, industrial networks, automotive networks, and many other applications. It can provide high precision time synchronization as fine as nano-seconds. The protocol depends on a Precision Time Protocol (PTP) engine to decide its own state automatically, and a PTP transportation layer to carry the PTP timing and various quality messages. The

configuration parameters and state data sets of IEEE 1588-2008 are numerous.

According to the concepts described in [RFC3444], IEEE 1588-2008 itself provides an information model in its normative specifications for the data sets (in IEEE 1588-2008 clause 8). Some standardization organizations including the IETF have specified data models in MIBs (Management Information Bases) for IEEE 1588-2008 data sets (e.g. [RFC8173], [IEEE8021AS]). These MIBs are typically focused on retrieval of state data using the Simple Network Management Protocol (SNMP), furthermore, configuration of PTP data sets is not considered in [RFC8173].

Some service providers and applications require that the management of the IEEE 1588-2008 synchronization network be flexible and more Internet-based (typically overlaid on their transport networks). Software Defined Network (SDN) is another driving factor, which demands an improved configuration capability of synchronization networks.

YANG [RFC7950] is a data modeling language used to model configuration and state data manipulated by network management protocols like the Network Configuration Protocol (NETCONF) [RFC6241]. A small set of built-in data types are defined in [RFC7950], and a collection of common data types are further defined in [RFC6991]. Advantages of YANG include Internet based configuration capability, validation, rollback and so on. All of these characteristics make it attractive to become another candidate modeling language for IEEE 1588-2008.

This document defines a YANG data model for the configuration of IEEE 1588-2008 devices and clocks, and retrieval of the state data of IEEE 1588-2008 clocks. The data model is based on the PTP data sets as specified in [IEEE1588]. The technology specific IEEE 1588-2008 information, e.g., those specifically implemented by a bridge, a router or a telecom profile, is out of scope of this document.

The YANG module in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

When used in practice, network products in support of synchronization typically conform to one or more IEEE 1588-2008 profiles. Each profile specifies how IEEE 1588-2008 is used in a given industry (e.g. telecom, automotive) and application. A profile can require features that are optional in IEEE 1588-2008, and it can specify new features that use IEEE 1588-2008 as a foundation.

It is expected that the IEEE 1588-2008 YANG module be used as follows:

o The IEEE 1588-2008 YANG module can be used as-is for products that conform to one of the default profiles specified in IEEE 1588-2008.

o When the IEEE 1588 standard is revised (e.g. the IEEE 1588 revision in progress at the time of writing this document), it will add some new optional features to its data sets. The YANG module of this document can be revised and extended to support these new features. Moreover, the YANG "revision" MUST be used to indicate changes to the YANG module under such a circumstance.

o A profile standard based on IEEE 1588-2008 may create a dedicated YANG module for its profile. The profile's YANG module SHOULD use YANG "import" to import the IEEE 1588-2008 YANG module as its foundation. Then the profile's YANG module SHOULD use YANG "augment" to add any profile-specific enhancements.

o A product that conforms to a profile standard may also create its own YANG module. The product's YANG module SHOULD "import" the profile's module, and then use YANG "augment" to add any product-specific enhancements.

1.1. Conventions used in this document

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.

1.2. Terminology

Most terminologies used in this document are extracted from [IEEE1588].

BC Boundary Clock, see Section 3.1.3 of [IEEE1588]

DS Data Set

E2E End-to-End

EUI Extended Unique Identifier

GPS Global Positioning System

IANA Internet Assigned Numbers Authority

IP Internet Protocol

NIST National Institute of Standards and Technology

NTP Network Time Protocol

OC Ordinary Clock, see Section 3.1.22 of [IEEE1588]

P2P Peer-to-Peer

PTP Precision Time Protocol

TAI International Atomic Time

TC Transparent Clock, see Section 3.1.46 of [IEEE1588]

UTC Coordinated Universal Time

PTP data set
: Structured attributes of clocks (an OC, BC or TC) used for PTP protocol decisions and for providing values for PTP message fields, see Section 8 of [IEEE1588].

PTP instance
: A PTP implementation in the device (i.e., an OC or BC) represented by a specific PTP data set.

2. IEEE 1588-2008 YANG Model hierarchy

This section describes the hierarchy of an IEEE 1588-2008 YANG module. Query and configuration of device wide or port specific configuration information and clock data set are described for this version.

Query and configuration of clock information include:

(Note: The attribute names are consistent with IEEE 1588-2008, but changed to the YANG style, i.e., using all lower-case, with dashes between words.)

- Clock data set attributes in a clock node, including: current-ds, parent-ds, default-ds, time-properties-ds, and transparent-clock-default-ds.

- Port-specific data set attributes, including: port-ds and transparent-clock-port-ds.

The readers are assumed to be familiar with IEEE 1588-2008. As all PTP terminologies and PTP data set attributes are described in details in IEEE 1588-2008 [IEEE1588], this document only outlines each of them in the YANG module.

A simplified YANG tree diagram [RFC8340] representing the data model is typically used by YANG modules. This document uses the same tree diagram syntax as described in [RFC8340].

```
module: ietf-ptp
  +--rw ptp
     +--rw instance-list* [instance-number]
        |  +--rw instance-number      uint32
        |  +--rw default-ds
        |  |  +--rw two-step-flag?    boolean
        |  |  +--ro clock-identity?   clock-identity-type
        |  |  +--rw number-ports?     uint16
        |  |  +--rw clock-quality
        |  |  |  +--rw clock-class?                 uint8
        |  |  |  +--rw clock-accuracy?              uint8
        |  |  |  +--rw offset-scaled-log-variance?  uint16
        |  |  +--rw priority1?        uint8
        |  |  +--rw priority2?        uint8
        |  |  +--rw domain-number?    uint8
        |  |  +--rw slave-only?       boolean
        |  +--rw current-ds
        |  |  +--rw steps-removed?        uint16
        |  |  +--rw offset-from-master?   time-interval-type
        |  |  +--rw mean-path-delay?      time-interval-type
        |  +--rw parent-ds
        |  |  +--rw parent-port-identity
        |  |  |  +--rw clock-identity?   clock-identity-type
        |  |  |  +--rw port-number?      uint16
        |  |  +--rw parent-stats?                  boolean
        |  |  +--rw observed-parent-offset-scaled-log-variance? uint16
        |  |  +--rw observed-parent-clock-phase-change-rate?    int32
        |  |  +--rw grandmaster-identity?         clock-identity-type
        |  |  +--rw grandmaster-clock-quality
        |  |  |  +--rw clock-class?                 uint8
        |  |  |  +--rw clock-accuracy?              uint8
        |  |  |  +--rw offset-scaled-log-variance?  uint16
        |  |  +--rw grandmaster-priority1?          uint8
```

```
|  |  +--rw grandmaster-priority2?           uint8
|  +--rw time-properties-ds
|  |  +--rw current-utc-offset-valid?   boolean
|  |  +--rw current-utc-offset?         int16
|  |  +--rw leap59?                     boolean
|  |  +--rw leap61?                     boolean
|  |  +--rw time-traceable?             boolean
|  |  +--rw frequency-traceable?        boolean
|  |  +--rw ptp-timescale?              boolean
|  |  +--rw time-source?                uint8
|  +--rw port-ds-list* [port-number]
|     +--rw port-number              uint16
|     +--rw port-state?              port-state-enumeration
|     +--rw underlying-interface?         if:interface-ref
|     +--rw log-min-delay-req-interval?   int8
|     +--rw peer-mean-path-delay?         time-interval-type
|     +--rw log-announce-interval?        int8
|     +--rw announce-receipt-timeout?     uint8
|     +--rw log-sync-interval?            int8
|     +--rw delay-mechanism?       delay-mechanism-enumeration
|     +--rw log-min-pdelay-req-interval?   int8
|     +--rw version-number?                uint8
+--rw transparent-clock-default-ds
|  +--ro clock-identity?    clock-identity-type
|  +--rw number-ports?      uint16
|  +--rw delay-mechanism?   delay-mechanism-enumeration
|  +--rw primary-domain?    uint8
+--rw transparent-clock-port-ds-list* [port-number]
   +--rw port-number                     uint16
   +--rw log-min-pdelay-req-interval?   int8
   +--rw faulty-flag?                    boolean
   +--rw peer-mean-path-delay?          time-interval-type
```

2.1. Interpretations from IEEE 1588 Working Group

The preceding model and the associated YANG module have some subtle differences from the data set specifications of IEEE Std 1588-2008. These differences are based on interpretation from the IEEE 1588 Working Group, and are intended to provide compatibility with future revisions of the IEEE 1588 standard.

In IEEE Std 1588-2008, a physical product can implement multiple PTP clocks (i.e., ordinary, boundary, or transparent clock). As specified in 1588-2008 subclause 7.1, each of the multiple clocks operates in an independent domain. However, the organization of multiple PTP domains was not clear in the data sets of IEEE Std 1588-2008. This document introduces the concept of PTP instance as described in the new revision of IEEE 1588. The instance concept is used exclusively to allow for optional support of multiple domains. The instance number has no usage within PTP messages.

Based on statements in IEEE 1588-2008 subclauses 8.3.1 and 10.1, most transparent clock products have interpreted the transparent clock data sets to reside as a singleton at the root level of the managed product, and this YANG model reflects that location.

2.2. Configuration and state

The information model of IEEE Std 1588-2008 classifies each member in PTP data sets as one of the following:

- Configurable: Writable by management.

- Dynamic: Read-only to management, and the value is changed by 1588 protocol operation.

- Static: Read-only to management, and the value typically does not change.

For details on the classification of each PTP data set member, refer to the IEEE Std 1588-2008 specification for that member.

Under certain circumstances, the classification of an IEEE 1588 data set member may change for a YANG implementation, for example, a configurable member needs to be changed to read-only. In such a case, an implementation SHOULD choose to return a warning upon writing to a read-only member, or use the deviation mechanism to develop a new deviation model as described in Section 7.20.3 of [RFC7950].

3. IEEE 1588-2008 YANG Module

This module imports typedef "interface-ref" from [RFC8343]. Most attributes are based on the information model defined in [IEEE1588], but their names are adapted to the YANG style of naming.

```
<CODE BEGINS> file "ietf-ptp@2018-09-10.yang"
//Note to RFC Editor: update the date to date of publication
module ietf-ptp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-ptp";
  prefix "ptp";

  import ietf-interfaces {
    prefix if;
    reference
       "RFC8343: A YANG Data Model for Interface Management";
  }

  organization "IETF TICTOC Working Group";
  contact
      "WG Web:   http://tools.ietf.org/wg/tictoc/
       WG List:  <mailto:tictoc@ietf.org>
       Editor:   Yuanlong Jiang
                 <mailto:jiangyuanlong@huawei.com>
       Editor:   Rodney Cummings
                 <mailto:rodney.cummings@ni.com>";
  description
    "This YANG module defines a data model for the configuration
     of IEEE 1588-2008 clocks, and also for retrieval of the state
     data of IEEE 1588-2008 clocks.";

  revision "2018-09-10" {
    //Note to RFC Editor: update the date to date of publication
    description "Initial version";
    reference "RFC XXXX: YANG Data Model for IEEE 1588-2008";
    //Note to RFC Editor: update RFC XXXX to the actual RFC number

  }

  typedef delay-mechanism-enumeration {
    type enumeration {
      enum e2e {
        value 1;
        description
          "The port uses the delay request-response mechanism.";
      }
```

```
        enum p2p {
          value 2;
          description
            "The port uses the peer delay mechanism.";
        }
        enum disabled {
          value 254;
          description
            "The port does not implement any delay mechanism.";
        }
      }
      description
        "The propagation delay measuring option used by the
         port. Values for this enumeration are specified
         by the IEEE 1588 standard exclusively.";
      reference
        "IEEE Std 1588-2008: 8.2.5.4.4";
    }

    typedef port-state-enumeration {
      type enumeration {
        enum initializing {
          value 1;
          description
            "The port is initializing its data sets, hardware, and
             communication facilities.";
        }
        enum faulty {
          value 2;
          description
            "The port is in the fault state.";
        }
        enum disabled {
          value 3;
          description
            "The port is disabled, and is not communicating PTP
             messages (other than possibly PTP management
             messages).";
        }
        enum listening {
          value 4;
          description
            "The port is listening for an Announce message.";
        }
        enum pre-master {
          value 5;
          description
```

```
            "The port is in the pre-master state.";
          }
          enum master {
            value 6;
            description
              "The port is behaving as a master port.";
          }
          enum passive {
            value 7;
            description
              "The port is in the passive state.";
          }
          enum uncalibrated {
            value 8;
            description
              "A master port has been selected, but the port is still
               in the uncalibrated state.";
          }
          enum slave {
            value 9;
            description
              "The port is synchronizing to the selected master port.";
          }
        }

        description
          "The current state of the protocol engine associated
           with the port.  Values for this enumeration are specified
           by the IEEE 1588 standard exclusively.";
        reference
          "IEEE Std 1588-2008: 8.2.5.3.1, 9.2.5";
      }

      typedef time-interval-type {
        type int64;
        description
          "Derived data type for time interval, represented in units of
            nanoseconds and multiplied by 2^16";
        reference
          "IEEE Std 1588-2008: 5.3.2";
      }

      typedef clock-identity-type {
        type binary {
          length "8";
        }
        description
```

```
        "Derived data type to identify a clock";
      reference
        "IEEE Std 1588-2008: 5.3.4";
    }

    grouping clock-quality-grouping {
      description
        "Derived data type for quality of a clock, which contains
         clockClass, clockAccuracy and offsetScaledLogVariance.";
      reference
        "IEEE Std 1588-2008: 5.3.7";

      leaf clock-class {
        type uint8;
        default 248;
        description
          "The clockClass denotes the traceability of the time
           or frequency distributed by the clock.";
      }

      leaf clock-accuracy {
        type uint8;
        description
          "The clockAccuracy indicates the expected accuracy
           of the clock.";
      }

      leaf offset-scaled-log-variance {
        type uint16;
        description
          "The offsetScaledLogVariance provides an estimate of
           the variations of the clock from a linear timescale
           when it is not synchronized to another clock
           using the protocol.";
      }
    }


    container ptp {
      description
        "The PTP struct containing all attributes of PTP data set,
          other optional PTP attributes can be augmented as well.";

      list instance-list {

        key "instance-number";
```

```
        description
          "List of one or more PTP data sets in the device (see IEEE
           Std 1588-2008 subclause 6.3).
           Each PTP data set represents a distinct instance of
           PTP implementation in the device (i.e., distinct
           Ordinary Clock or Boundary Clock).";

        leaf instance-number {
          type uint32;
          description
             "The instance number of the current PTP instance.
              This instance number is used for management purposes
              only. This instance number does not represent the PTP
              domain number, and is not used in PTP messages.";
        }

        container default-ds {
          description
            "The default data set of the clock (see IEEE Std
             1588-2008 subclause 8.2.1). This data set represents
             the configuration/state required for operation
             of Precision Time Protocol (PTP) state machines.";

          leaf two-step-flag {
            type boolean;
            description
              "When set to true, the clock is a two-step clock;
               otherwise,the clock is a one-step clock.";
          }

          leaf clock-identity {
            type clock-identity-type;
            config false;
            description
              "The clockIdentity of the local clock";
          }

          leaf number-ports {
            type uint16;
            description
              "The number of PTP ports on the instance.";
          }

          container clock-quality {
            description
              "The clockQuality of the local clock.";
```

```
            uses clock-quality-grouping;
          }

          leaf priority1 {
            type uint8;
            description
              "The priority1 attribute of the local clock.";
          }

          leaf priority2{
            type uint8;
            description
              "The priority2 attribute of the local clock.";
          }

          leaf domain-number {
            type uint8;
            description
              "The domain number of the current syntonization
               domain.";
          }

          leaf slave-only {
            type boolean;
            description
              "When set to true, the clock is a slave-only clock.";
          }

        }

        container current-ds {
          description
            "The current data set of the clock (see IEEE Std
             1588-2008 subclause 8.2.2). This data set represents
             local states learned from the exchange of
             Precision Time Protocol (PTP) messages.";

          leaf steps-removed {
            type uint16;
            default 0;
            description
              "The number of communication paths traversed
               between the local clock and the grandmaster clock.";
          }

          leaf offset-from-master {
            type time-interval-type;
```

```
            description
              "The current value of the time difference between
               a master and a slave clock as computed by the slave.";
            }

            leaf mean-path-delay {
              type time-interval-type;
              description
                "The current value of the mean propagation time between
                 a master and a slave clock as computed by the slave.";

            }

          }

          container parent-ds {
            description
              "The parent data set of the clock (see IEEE Std 1588-2008
               subclause 8.2.3).";

            container parent-port-identity {
              description
                "The portIdentity of the port on the master, it
                 contains two members: clockIdentity and portNumber.";
              reference
                "IEEE Std 1588-2008: 5.3.5";

              leaf clock-identity {
                type clock-identity-type;
                description
                  "Identity of the clock";
              }

              leaf port-number {
                type uint16;
                description
                  "Port number";
              }
            }

            leaf parent-stats {
              type boolean;
              default false;
              description
                "When set to true, the values of
                 observedParentOffsetScaledLogVariance and
                 observedParentClockPhaseChangeRate of parentDS
```

```
            have been measured and are valid.";
        }

        leaf observed-parent-offset-scaled-log-variance {
          type uint16;
          default 65535;
          description
            "An estimate of the parent clock's PTP variance
             as observed by the slave clock.";
        }

        leaf observed-parent-clock-phase-change-rate {
          type int32;
          description
            "An estimate of the parent clock's phase change rate
             as observed by the slave clock.";
        }

        leaf grandmaster-identity {
          type clock-identity-type;
          description
            "The clockIdentity attribute of the grandmaster clock.";
        }

        container grandmaster-clock-quality {
          description
            "The clockQuality of the grandmaster clock.";
          uses clock-quality-grouping;
        }

        leaf grandmaster-priority1 {
          type uint8;
          description
            "The priority1 attribute of the grandmaster clock.";
        }

        leaf grandmaster-priority2 {
          type uint8;
          description
            "The priority2 attribute of the grandmaster clock.";
        }

      }

      container time-properties-ds {
        description
          "The timeProperties data set of the clock (see
```

```
           IEEE Std 1588-2008 subclause 8.2.4).";

        leaf current-utc-offset-valid {
          type boolean;
          description
            "When set to true, the current UTC offset is valid.";
        }
        leaf current-utc-offset {
          when "../current-utc-offset-valid='true'";
          type int16;
          description
            "The offset between TAI and UTC when the epoch of the
             PTP system is the PTP epoch in units of seconds, i.e.,
             when ptp-timescale is TRUE; otherwise, the value has
             no meaning.";
        }


        leaf leap59 {
          type boolean;
          description
            "When set to true, the last minute of the current UTC
             day contains 59 seconds.";
        }

        leaf leap61 {
          type boolean;
          description
            "When set to true, the last minute of the current UTC
             day contains 61 seconds.";
        }

        leaf time-traceable {
          type boolean;
          description
            "When set to true, the timescale and the
             currentUtcOffset are traceable to a primary
             reference.";
        }

        leaf frequency-traceable {
          type boolean;
          description
            "When set to true, the frequency determining the
             timescale is traceable to a primary reference.";
        }
```

```
          leaf ptp-timescale {
            type boolean;
            description
              "When set to true, the clock timescale of the
               grandmaster clock is PTP; otherwise, the timescale is
               ARB
               (arbitrary).";
          }

          leaf time-source {
            type uint8;
            description
              "The source of time used by the grandmaster clock.";
          }
        }

        list port-ds-list {
          key "port-number";
          description
            "List of port data sets of the clock (see IEEE Std
             1588-2008 subclause 8.2.5).";

          leaf port-number {
            type uint16;

            description
              "Port number.
               The data sets (i.e., information model) of IEEE Std
               1588-2008 specify a member portDS.portIdentity, which
               uses a typed struct with members clockIdentity and
               portNumber.

               In this YANG data model, portIdentity is not modeled
               in the port-ds-list, however, its members are provided
               as follows:
               portIdentity.portNumber is provided as this port-
               number leaf in port-ds-list; and
               portIdentity.clockIdentity is provided as the clock-
               identity leaf in default-ds of the instance
               (i.e., ../../default-ds/clock-identity).";
          }

          leaf port-state {
            type port-state-enumeration;
            default "initializing";
            description
              "Current state associated with the port.";
```

```
          }

          leaf underlying-interface {
           type if:interface-ref;
           description
              "Reference to the configured underlying interface that
               is used by this PTP Port (see RFC 8343).";
          }

          leaf log-min-delay-req-interval {
            type int8;
            description
              "The base-two logarithm of the minDelayReqInterval
               (the minimum permitted mean time interval between
               successive Delay_Req messages).";
          }

          leaf peer-mean-path-delay {
            type time-interval-type;
            default 0;
            description
              "An estimate of the current one-way propagation delay
               on the link when the delayMechanism is P2P; otherwise,
               it is zero.";
          }

          leaf log-announce-interval {
            type int8;
            description
              "The base-two logarithm of the mean
               announceInterval (mean time interval between
               successive Announce messages).";
          }

          leaf announce-receipt-timeout {
            type uint8;
            description
              "The number of announceInterval that have to pass
               without receipt of an Announce message before the
               occurrence of the event ANNOUNCE_RECEIPT_TIMEOUT_
               EXPIRES.";
          }

          leaf log-sync-interval {
            type int8;
            description
              "The base-two logarithm of the mean SyncInterval
```

```
                for multicast messages.  The rates for unicast
                transmissions are negotiated separately on a per port
                basis and are not constrained by this attribute.";
            }

            leaf delay-mechanism {
              type delay-mechanism-enumeration;
              description
                "The propagation delay measuring option used by the
                 port in computing meanPathDelay.";
            }

            leaf log-min-pdelay-req-interval {
              type int8;
              description
                "The base-two logarithm of the
                 minPdelayReqInterval (minimum permitted mean time
                 interval between successive Pdelay_Req messages).";

            }

            leaf version-number {
              type uint8;
              description
                "The PTP version in use on the port.";
            }

          }
        }

        container transparent-clock-default-ds {
          description
            "The members of the transparentClockDefault data set (see
             IEEE Std 1588-2008 subclause 8.3.2).";

          leaf clock-identity {
            type clock-identity-type;
            config false;
            description
              "The clockIdentity of the transparent clock.";
          }

          leaf number-ports {
            type uint16;
            description
              "The number of PTP ports on the transparent clock.";
          }
```

```
        leaf delay-mechanism {
          type delay-mechanism-enumeration;
          description
            "The propagation delay measuring option
             used by the transparent clock.";
        }

        leaf primary-domain {
          type uint8;
          default 0;
          description
            "The domainNumber of the primary syntonization domain (see
             IEEE Std 1588-2008 subclause 10.1).";
        }
      }

      list transparent-clock-port-ds-list {
        key "port-number";
        description
          "List of transparentClockPort data sets of the transparent
           clock (see IEEE Std 1588-2008 subclause 8.3.3).";

        leaf port-number {
            type uint16;
            description
              "Port number.
               The data sets (i.e., information model) of IEEE Std
               1588-2008 specify a member
               transparentClockPortDS.portIdentity, which uses a typed
               struct with members clockIdentity and portNumber.

               In this YANG data model, portIdentity is not modeled in
               the transparent-clock-port-ds-list, however, its
               members are provided as follows:
               portIdentity.portNumber is provided as this leaf member
               in transparent-clock-port-ds-list; and
               portIdentity.clockIdentity is provided as the clock-
               identity leaf in transparent-clock-default-ds
               (i.e., ../../transparent-clock-default-ds/clock-
               identity).";

        }


        leaf log-min-pdelay-req-interval {
          type int8;
```

```
          description
            "The logarithm to the base 2 of the
             minPdelayReqInterval (minimum permitted mean time
             interval between successive Pdelay_Req messages).";
        }

        leaf faulty-flag {
          type boolean;
          default false;
          description
            "When set to true, the port is faulty.";
        }

        leaf peer-mean-path-delay {
          type time-interval-type;
          default 0;
          description
            "An estimate of the current one-way propagation delay
             on the link when the delayMechanism is P2P; otherwise,
             it is zero.";
        }

       }
     }
   }
```

<CODE ENDS>

4. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446]. Furthermore, general security considerations of time protocols are discussed in [RFC7384].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module are writable, and the involved subtrees that are sensitive include:

/ptp/instance-list specifies an instance (i.e., PTP data sets) for an OC or BC.

/ptp/transparent-clock-default-ds specifies a default data set for a TC.

/ptp/transparent-clock-port-ds-list specifies a list of port data sets for a TC.

Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. Specifically, an inappropriate configuration of them may adversely impact a PTP synchronization network. For example, loss of synchronization on a clock, accuracy degradation on a set of clocks, or even break down of a whole synchronization network.

5. IANA Considerations

This document registers the following URI in the "IETF XML registry" [RFC3688]:
URI: urn:ietf:params:xml:ns:yang:ietf-ptp
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace

This document registers the following YANG module in the "YANG Module Names" registry [RFC6020]:
Name: ietf-ptp
Namespace: urn:ietf:params:xml:ns:yang:ietf-ptp
Prefix: ptp
Reference: RFC XXXX

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997

[RFC3688] Mealling, M., "The IETF XML Registry", RFC 3688, January 2004

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF) ", RFC 6020, October 2010

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and Bierman, A., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013

[RFC7950] Bjorklund, M., "The YANG 1.1 Data Modeling Language", RFC 7950, August 2016

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, January 2017

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017

[RFC8341] Bierman, A. and Bjorklund, M., "Network Configuration Protocol (NETCONF) Access Control Model", RFC 8341, March 2018

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, March 2018

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, March 2018

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, August 2018

[IEEE1588] IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, July 2008

6.2. Informative References

[IEEE8021AS] IEEE, "Timing and Synchronizations for Time-Sensitive Applications in Bridged Local Area Networks", IEEE 802.1AS-2001, 2011

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, January 2003

[RFC4663] Harrington, D., "Transferring MIB Work from IETF Bridge MIB WG to IEEE 802.1 WG", RFC 4663, September 2006

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, October 2014

[RFC8340] Bjorklund, M., and Berger, L., "YANG Tree Diagrams", RFC 8340, March 2018

[RFC8173] Shankarkumar, V., Montini, L., Frost, T., and Dowd, G., "Precision Time Protocol Version 2 (PTPv2) Management Information Base", RFC 8173, June 2017

7. Acknowledgments

The authors would like to thank Tom Petch, Radek Krejci, Mahesh Jethanandani, Tal Mizrahi, Opher Ronen, Liang Geng, Alex Campbell, Joe Gwinn, John Fletcher, William Zhao and Dave Thaler for their valuable reviews and suggestions, thank Benoit Claise and Radek Krejci for their validation of the YANG module, and thank Jingfei Lv and Zitao Wang for their discussions on IEEE 1588 and YANG respectively.

Appendix A Transferring YANG Work to IEEE 1588 WG

This Appendix is informational.

This appendix describes a future plan to transition responsibility for IEEE 1588 YANG modules from the IETF TICTOC Working Group (WG) to the IEEE 1588 WG, which develops the time synchronization technology that the YANG modules are designed to manage.

This appendix is forward-looking with regard to future standardization roadmaps in IETF and IEEE. Since those roadmaps cannot be predicted with significant accuracy, this appendix is informational, and it does not specify imperatives or normative specifications of any kind.

The IEEE 1588-2008 YANG module of this standard represents a cooperation between IETF (for YANG) and IEEE (for 1588). For the initial standardization of IEEE-1588 YANG modules, the information model is relatively clear (i.e., IEEE 1588 data sets), but expertise in YANG is required, making IETF an appropriate location for the standards. The TICTOC WG has expertise with IEEE 1588, making it the appropriate location within IETF.

The IEEE 1588 WG anticipates future changes to its standard on an ongoing basis. As IEEE 1588 WG members gain practical expertise with YANG, the IEEE 1588 WG will become more appropriate for standardization of its YANG modules. As the IEEE 1588 standard is revised and/or amended, IEEE 1588 members can more effectively synchronize the revision of this YANG module with future versions of the IEEE 1588 standard.

This appendix is meant to establish some clear expectations between IETF and IEEE about the future transfer of IEEE 1588 YANG modules to the IEEE 1588 WG. The goal is to assist in making the future transfer as smooth as possible. As the transfer takes place, some case-by-case situations are likely to arise, which can be handled by discussion on the IETF TICTOC WG mailing lists and/or appropriate liaisons.

This appendix obtained insight from [RFC4663], an informational memo that described a similar transfer of MIB work from the IETF Bridge MIB WG to the IEEE 802.1 WG.

A.1. Assumptions for the Transfer

For the purposes of discussion in this appendix, assume that the IESG has approved the publication of an RFC containing a YANG module for a published IEEE 1588 standard. As of this writing, this is IEEE Std 1588-2008, but it is possible that YANG modules for subsequent 1588 revisions could be published from the IETF TICTOC WG. For discussion in this appendix, we use the phrase "last IETF 1588 YANG" to refer to the most recently published 1588 YANG module from the IETF TICTOC WG.

The IEEE-SA Standards Board New Standards Committee (NesCom) handles new Project Authorization Requests (PARs) (see http://standards.ieee.org/board/nes/). PARs are roughly the equivalent of IETF Working Group Charters and include information concerning the scope, purpose, and justification for standardization projects.

Assume that IEEE 1588 has an approved PAR that explicitly specifies development of a YANG module. The transfer of YANG work will occur in the context of this IEEE 1588 PAR. For discussion in this appendix, we use the phrase "first IEEE 1588 YANG" to refer to the first IEEE 1588 standard for YANG.

Assume that as part of the transfer of YANG work, the IETF TICTOC WG agrees to cease all work on standard YANG modules for IEEE 1588.

Assume that the IEEE 1588 WG has participated in the development of the last IETF 1588 YANG module, such that the first IEEE 1588 YANG module will effectively be a revision of it. In other words, the transfer of YANG work will be relatively clean.

The actual conditions for the future transfer can be such that the preceding assumptions do not hold. Exceptions to the assumptions will need to be addressed on a case-by-case basis at the time of the transfer. This appendix describes topics that can be addressed based on the preceding assumptions.

A.2. Intellectual Property Considerations

During review of the legal issues associated with transferring Bridge MIB WG documents to the IEEE 802.1 WG (Section 3.1 and Section 9 of [RFC4663]), it was concluded that the IETF does not have sufficient legal authority to make the transfer to IEEE without the consent of the document authors.

If the last IETF 1588 YANG is published as a RFC, the work is required to be transferred from the IETF to the IEEE, so that IEEE 1588 WG can begin working on the first IEEE 1588 YANG.

When work on the first IEEE YANG module begins in the IEEE 1588 WG, that work derives from the last IETF YANG module of this RFC, requiring a transfer of that work from the IETF to the IEEE. In order to avoid having the transfer of that work be dependent on the availability of this RFC's authors at the time of its publication, the IEEE Standards Association department of Risk Management and Licensing provided the appropriate forms and mechanisms for this document's authors to assign a non-exclusive license for IEEE to create derivative works from this document. Those IEEE forms and mechanisms will be updated as needed for any future IETF YANG modules for IEEE 1588 (The signed forms are held by the IEEE Standards Association department of Risk Management and Licensing.). This will help to make the future transfer of work from IETF to IEEE occur as smoothly as possible.

As stated in the initial "Status of this Memo", the YANG module in this document conforms to the provisions of BCP 78. The IETF will retain all the rights granted at the time of publication in the published RFCs.

A.3. Namespace and Module Name

As specified in Section 5 "IANA Considerations", the YANG module in this document uses IETF as the root of its URN namespace and YANG module name.

Use of IETF as the root of these names implies that the YANG module is standardized in a Working Group of IETF, using the IETF processes. If the IEEE 1588 Working Group were to continue using these names rooted in IETF, the IEEE 1588 YANG standardization would need to continue in the IETF. The goal of transferring the YANG work is to avoid this sort of dependency between standards organizations.

IEEE 802 has an active PAR (IEEE P802d) for creating a URN namespace for IEEE use (see http://standards.ieee.org/develop/project/802d.html). It is likely that this IEEE 802 PAR will be approved and published prior to the transfer of YANG work to the IEEE 1588 WG. If so, the IEEE 1588 WG can use the IEEE URN namespace for the first IEEE 1588 YANG module, such as:

```
urn:ieee:Std:1588:yang:ieee1588-ptp
```

where "ieee1588-ptp" is the registered YANG module name in the IEEE.

Under the assumptions of section A.1, the first IEEE 1588 YANG module's prefix will be the same as the last IETF 1588 YANG module's prefix (i.e. "ptp"). Consequently, other YANG modules can preserve the same import prefix "ptp" to access PTP nodes during the migration from the last IETF 1588 YANG module to the first IEEE 1588 YANG module.

The result of these name changes are that for complete compatibility, a server (i.e., IEEE 1588 node) can choose to implement a YANG module for the last IETF 1588 YANG module (with IETF root) as well as the first IEEE 1588 YANG module (with IEEE root). Since the content of the YANG module transferred are the same, the server implementation is effectively common for both.

From a client's perspective, a client of the last IETF 1588 YANG module (or earlier) looks for the IETF-rooted module name; and a client of the first IEEE 1588 YANG module (or later) looks for the IEEE-rooted module name.

A.4. IEEE 1588 YANG Modules in ASCII Format

Although IEEE 1588 can certainly decide to publish YANG modules only in the PDF format that they use for their standard documents, without publishing an ASCII version, most network management systems cannot import the YANG module directly from the PDF. Thus, not publishing an ASCII version of the YANG module would negatively impact implementers and deployers of YANG modules and would make potential IETF reviews of YANG modules more difficult.

This appendix recommends that the IEEE 1588 WG consider future plans for:

- Public availability of the ASCII YANG modules during project development. These ASCII files allow IETF participants to access these documents for pre-standard review purposes.

- Public availability of the YANG portion of published IEEE 1588 standards, provided as an ASCII file for each YANG module. These ASCII files are intended for use of the published IEEE 1588 standard.

As an example of public availability during project development, IEEE 802 uses the same repository that IETF uses for YANG module development (see https://github.com/YangModels/yang). IEEE branches are provided for experimental work (i.e. pre-PAR) as well as

standard work (post-PAR drafts). IEEE-SA has approved use of this repository for project development, but not for published standards.

As an example of public availability of YANG modules for published standards, IEEE 802.1 provides a public list of ASCII files for MIB (see http://www.ieee802.org/1/files/public/MIBs/ and http://www.ieee802.org/1/pages/MIBS.html), and analogous lists are planned for IEEE 802.1 YANG files.

Authors' Addresses

Yuanlong Jiang (Editor)
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129, China
Email: jiangyuanlong@huawei.com

Xian Liu
Independent
Shenzhen 518129, China
lene.liuxian@foxmail.com

Jinchun Xu
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129, China
xujinchun@huawei.com

Rodney Cummings (Editor)
National Instruments
11500 N. Mopac Expwy
Bldg. C
Austin, TX 78759-3504
Email: Rodney.Cummings@ni.com

TICTOC Working Group Doug Arnold
Internet Draft Meinberg-USA
Intended status: Standards Track Heiko Gerstung
Meinberg
Expires: July 22, 2016 Jan 22, 2016

Enterprise Profile for the Precision Time Protocol With Mixed Multicast and Unicast Messages

draft-ietf-tictoc-ptp-enterprise-profile-06.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on Agust 4, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

This document describes a profile for the use of the Precision Time Protocol in an IPV4 or IPv6 Enterprise information system environment. The profile uses the End to End Delay Measurement Mechanism, allows both multicast and unicast Delay Request and Delay Response Messages.

Table of Contents

1. Introduction 2
2. Conventions used in this document 3
3. Technical Terms 3
4. Problem Statement 5
5. Network Technology 6
6. Time Transfer and Delay Measurement 7
7. Default Message Rates 8
8. Requirements for Master Clocks 8
9. Requirements for Slave Clocks 9
10. Requirements for Transparent Clocks 9
11. Requirements for Boundary Clocks 10
12. Management and Signaling Messages 10
13. Forbidden PTP Options 10
14. Interoperation with Other PTP Profiles 10
15. Security Considerations 11
16. IANA Considerations 11
17. References 11
17.1. Normative References 11
17.2. Informative References 12
18. Acknowledgments 12
19. Authors addresses 12

1. Introduction

The Precision Time Protocol ("PTP"), standardized in IEEE 1588, has been designed in its first version (IEEE 1588-2002) with the goal to minimize configuration on the participating nodes. Network communication was based solely on multicast messages, which unlike NTP did not require that a receiving node ("slave clock") in [IEEE1588] needs to know the identity of the time sources in the network (the Master Clocks).

The so-called "Best Master Clock Algorithm" ([IEEE1588] Clause 9.3), a mechanism that all participating PTP nodes must follow, set up strict rules for all members of a PTP domain to determine which node shall be the active sending time source (Master Clock). Although the multicast communication model has advantages in smaller networks, it complicated the application of PTP in larger networks, for example in environments like IP based telecommunication networks or financial data centers. It is considered inefficient that, even if the content of a message applies only to one receiver, it is forwarded by the underlying

network (IP) to all nodes, requiring them to spend network bandwidth and other resources like CPU cycles to drop the message.

The second revision of the standard (IEEE 1588-2008) is the current version (also known as PTPv2) and introduced the possibility to use unicast communication between the PTP nodes in order to overcome the limitation of using multicast messages for the bi-directional information exchange between PTP nodes. The unicast approach avoided that, in PTP domains with a lot of nodes, devices had to throw away up to 99% of the received multicast messages because they carried information for some other node. PTPv2 also introduced so-called "PTP profiles" ([IEEE1588] Clause 19.3). This construct allows organizations to specify selections of attribute values and optional features, simplifying the configuration of PTP nodes for a specific application. Instead of having to go through all possible parameters and configuration options and individually set them up, selecting a profile on a PTP node will set all the parameters that are specified in the profile to a defined value. If a PTP profile definition allows multiple values for a parameter, selection of the profile will set the profile-specific default value for this parameter. Parameters not allowing multiple values are set to the value defined in the PTP profile. A number of PTP features and functions are optional and a profile should also define which optional features of PTP are required, permitted or prohibited. It is possible to extend the PTP standard with a PTP profile by using the TLV mechanism of PTP (see [IEEE1588] Clause 13.4), defining an optional Best Master Clock Algorithm and a few other ways. PTP has its own management protocol (defined in [IEEE1588] Clause 15.2) but allows a PTP profile specify an alternative management mechanism, for example SNMP.

2. Conventions used in this document

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].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

3. Technical Terms

Acceptable Master Table: A PTP Slave Clock may maintain a list of masters which it is willing to synchronize to.

Alternate Master: A PTP Master Clock, which is not the Best Master, may act as a master with the Alternate Master flag set on the messages it sends.

Announce message: Contains the master clock properties of a Master clock. Used to determine the Best Master.

Best Master: A clock with a port in the master state, operating consistently with the Best Master Clock Algorithm.

Best Master Clock Algorithm: A method for determining which state a port of a PTP clock should be in. The algorithm works by identifying which of several PTP Master capable clocks is the best master. Clocks have priority to become the acting Grandmaster, based on the properties each Master Clock sends in its Announce Message.

Boundary Clock: A device with more than one PTP port. Generally boundary clocks will have one port in slave state to receive timing and then other ports in master state to re-distribute the timing.

Clock Identity: In IEEE 1588-2008 this is a 64-bit number assigned to each PTP clock which must be unique. Often the Ethernet MAC address is used since there is already an international infrastructure for assigning unique numbers to each device manufactured.

Domain: Every PTP message contains a domain number. Domains are treated as separate PTP systems in the network. Slaves, however, can combine the timing information derived from multiple domains.

End to End Delay Measurement Mechanism: A network delay measurement mechanism in PTP facilitated by an exchange of messages between a Master Clock and Slave Clock.

Grandmaster: the primary master clock within a domain of a PTP system

IEEE 1588: The timing and synchronization standard which defines PTP, and describes The node, system, and communication properties necessary to support PTP.

Master clock: a clock with at least one port in the master state.

NTP: Network Time Protocol, defined by RFC 5905, see [NTP].

Ordinary Clock: A clock that has a single Precision Time Protocol (PTP) port in a domain and maintains the timescale used in the domain. It may serve as a master clock, or be a slave clock.

Peer to Peer Delay Measurement Mechanism: A network delay measurement mechanism in PTP facilitated by an exchange of messages between adjacent devices in a network.

Preferred Master: A device intended to act primarily as the Grandmaster of a PTP system, or as a back up to a Grandmaster.

PTP: The Precision Time Protocol, the timing and synchronization protocol define by IEEE 1588.

PTP port: An interface of a PTP clock with the network. Note that there may be multiple PTP ports running on one physical interface, for example a unicast slave which talks to several Grandmaster clocks in parallel.

PTPv2: Refers specifically to the second version of PTP defined by IEEE 1588-2008.

Rogue Master: A clock with a port in the master state, even though it should not be in the master state according to the Best Master Clock Algorithm, and does not set the alternate master flag.

Slave clock: a clock with at least one port in the slave state, and no ports in the master state.

Slave Only Clock: An Ordinary clock which cannot become a Master clock.

TLV: Type Length Value, a mechanism for extending messages in networked communications.

Transparent Clock. A device that measures the time taken for a PTP event message to transit the device and then updates the message with a correction for this transit time.

Unicast Discovery: A mechanism for PTP slaves to establish a unicast communication with PTP masters using a configures table of master IP addresses and Unicast Message Negotiation.

Unicast Negotiation: A mechanism in PTP for Slave Clocks to negotiate unicast Sync, announce and Delay Request Message Rates from a Master Clock.

4. Problem Statement

This document describes a version of PTP intended to work in large enterprise networks. Such networks are deployed, for example, in financial corporations. It is becoming increasingly common in such networks to perform distributed time tagged measurements, such as one-way packet latencies and cumulative delays on software systems spread across multiple computers. Furthermore there is often a desire to check the age of information time tagged by a different machine. To perform these measurements it is necessary to deliver a common precise time to multiple devices on a network. Accuracy currently required in the Financial Industry range from 100 microseconds to 500 nanoseconds to the Grandmaster. This profile does not specify timing performance requirements, but such requirements explain why the needs cannot always be met by NTP, as commonly implemented. Such accuracy cannot usually be achieved with a traditional time transfer such as NTP, without adding

non-standard customizations such as hardware time stamping, and on path support. These features are currently part of PTP, or are allowed by it. Because PTP has a complex range of features and options it is necessary to create a profile for enterprise networks to achieve interoperability between equipment manufactured by different vendors.

Although enterprise networks can be large, it is becoming increasingly common to deploy multicast protocols, even across multiple subnets. For this reason it is desired to make use of multicast whenever the information going to many destinations is the same. It is also advantageous to send information which is unique to one device as a unicast message. The latter can be essential as the number of PTP slaves becomes hundreds or thousands.

PTP devices operating in these networks need to be robust. This includes the ability to ignore PTP messages which can be identified as improper, and to have redundant sources of time.

5. Network Technology

This PTP profile SHALL operate only in networks characterized by UDP [RFC768] over either IPv4 [RFC791] or IPv6 [RFC2460], as described by Annexes D and E in [IEEE1588] respectively. If a network contains both IPv4 and IPv6, then they SHALL be treated as separate communication paths. Clocks which communicate using IPv4 can interact with clocks using IPv6 if there is an intermediary device which simultaneously communicates with both IP versions. A boundary clock might perform this function, for example. A PTP domain SHALL use either IPv4 or IPv6 over a communication path, but not both. The PTP system MAY include switches and routers. These devices MAY be transparent clocks, boundary clocks, or neither, in any combination. PTP Clocks MAY be Preferred Masters, Ordinary Clocks, or Boundary Clocks. The ordinary clocks may be Slave Only Clocks, or be master capable.

Note that clocks SHOULD always be identified by their clock ID and not the IP or Layer 2 address. This is important in IPv6 networks since Transparent clocks are required to change the source address of any packet which they alter. In IPv4 networks some clocks might be hidden behind a NAT, which hides their IP addresses from the rest of the network. Note also that the use of NATs may place limitations on the topology of PTP networks, depending on the port forwarding scheme employed. Details of implementing PTP with NATs are out of scope of this document.

Similar to NTP, PTP makes the assumption that the one way network delay for Sync Messages and Delay Response Messages are the same. When this is not true it can cause errors in the transfer of time from the Master to the Slave. It is up to the system integrator to design the network so that such effects do not prevent the PTP system from meeting the timing requirements. The details of

network asymmetry are outside the scope of this document. See for example, [G8271].

6. Time Transfer and Delay Measurement

Master clocks, Transparent clocks and Boundary clocks MAY be either one-step clocks or two-step clocks. Slave clocks MUST support both behaviors. The End to End Delay Measurement Method MUST be used.

Note that, in IP networks, Sync messages and Delay Request messages exchanged between a master and slave do not necessarily traverse the same physical path. Thus, wherever possible, the network SHOULD be traffic engineered so that the forward and reverse routes traverse the same physical path. Traffic engineering techniques for path consistency are out of scope of this document.

Sync messages MUST be sent as PTP event multicast messages (UDP port 319) to the PTP primary IP address. Two step clocks SHALL send Follow-up messages as PTP general messages (UDP port 320). Announce messages MUST be sent as multicast messages (UDP port 320) to the PTP primary address. The PTP primary IP address is 224.0.1.129 for IPv4 and FF0X:0:0:0:0:0:0:181 for Ipv6, where X can be a value between 0x0 and 0xF, see [IEEE1588] Annex E, Section E.3.

Delay Request Messages MAY be sent as either multicast or unicast PTP event messages. Master clocks SHALL respond to multicast Delay Request messages with multicast Delay Response PTP general messages. Master clocks SHALL respond to unicast Delay Request PTP event messages with unicast Delay Response PTP general messages. This allow for the use of Ordinary clocks which do not support the Enterprise Profile, as long as they are slave Only Clocks.

Clocks SHOULD include support for multiple domains. The purpose is to support multiple simultaneous masters for redundancy. Leaf devices (non-forwarding devices) can use timing information from multiple masters by combining information from multiple instantiations of a PTP stack, each operating in a different domain. Redundant sources of timing can be ensembled, and/or compared to check for faulty master clocks. The use of multiple simultaneous masters will help mitigate faulty masters reporting as healthy, network delay asymmetry, and security problems. Security problems include man-in-the-middle attacks such as delay attacks, packet interception / manipulation attacks. Assuming the path to each master is different, failures malicious or otherwise would have to happen at more than one path simultaneously. Whenever feasible, the underlying network transport technology SHOULD be configured so that timing messages in different domains traverse different network paths.

7. Default Message Rates

The Sync, Announce and Delay Request default message rates SHALL each be once per second. The Sync and Delay Request message rates MAY be set to other values, but not less than once every 128 seconds, and not more than 128 messages per second. The Announce message rate SHALL NOT be changed from the default value. The Announce Receipt Timeout Interval SHALL be three Announce Intervals for Preferred Masters, and four Announce Intervals for all other masters.

Unicast Discovery and Unicast Message Negotiation options MAY be utilized. If Unicat Negotiation is not used, operators will need to set a delay request interval, or make use of a default value.

8. Requirements for Master Clocks

Master clocks SHALL obey the standard Best Master Clock Algorithm from [IEEE1588]. PTP systems using this profile MAY support multiple simultaneous Grandmasters as long as each active Grandmaster is operating in a different PTP domain.

A port of a clock SHALL NOT be in the master state unless the clock has a current value for the number of UTC leap seconds.

9. Requirements for Slave Clocks

Slave clocks MUST be able to operate properly in a network which contains multiple Masters in multiple domains. Slaves SHOULD make use of information from the all Masters in their clock control subsystems. Slave Clocks MUST be able to operate properly in the presence of a Rogue Master. Slaves SHOULD NOT Synchronize to a Master which is not the Best Master in its domain. Slaves will continue to recognize a Best Master for the duration of the Announce Time Out Interval. Slaves MAY use an Acceptable Master Table. If a Master is not an Acceptable Master, then the Slave MUST NOT synchronize to it. Note that IEEE 1588-2008 requires slave clocks to support both two-step or one-step Master clocks. See [IEEE1588], section 11.2.

Since Announce messages are sent as multicast messages slaves can obtain the IP addresses of master from the Announce messages. Note that the IP source addresses of Sync and Follow-up messages may have been replaced by the source addresses of a transparent clock, so slaves MUST send Delay Request messages to the IP address in the Announce message. Sync and Follow-up messages can be correlated with the Announce message using the clock ID, which is never altered by Transparent clocks in this profile.

10. Requirements for Transparent Clocks

Transparent clocks SHALL NOT change the transmission mode of an Enterprise Profile PTP message. For example a Transparent clock SHALL NOT change a unicast message to a multicast message. Transparent Clocks SHOULD support multiple domains. Transparent Clocks which syntonize to the master clock will need to maintain separate clock rate offsets for each of the supported domains.

11. Requirements for Boundary Clocks

Boundary Clocks SHOULD support multiple simultaneous PTP domains. This will require them to maintain servo loops for each of the domains supported, at least in software. Boundary clocks MUST NOT combine timing information from different domains.

12. Management and Signaling Messages

PTP Management messages MAY be used. Any PTP management message which is sent with the targetPortIdentity.clockIdentity set to all 1s (all clocks) MUST be sent as a multicast message. Management messages with any other value of for the Clock Identity is intended for a specific clock and MUST be sent as a unicast message. Similarly, if any signaling messages are used they MUST also be sent as unicast messages whenever the message is intended for a specific clock.

13. Forbidden PTP Options

Clocks operating in the Enterprise Profile SHALL NOT use peer to peer timing for delay measurement. Grandmaster Clusters are NOT ALLOWED. The Alternate Master option is also forbidden. Clocks operating in the Enterprise Profile SHALL NOT use Alternate Timescales.

14. Interoperation with IEEE 1588 Default Profile

Clocks operating in the Enterprise Profile will interoperate with clocks operating in the Default Profile described in [IEEE1588] Annex J.3. This variant of the Default Profile uses the End to End Delay Measurement Mechanism. In addition the Default Profile would have to operates over IPv4 or IPv6 networks, and use management messages in unicast when those messages are directed at a specific clock. If either of these requirements are not met than Enterprise Profile clocks will not interoperate with Annex J.3 Default Profile Clocks. The Enterprise Profile Profile will will not interoperate with the Annex J.4 variant of the Default Profile which requires use of the Peer to Peer Delay Measurement Mechanism.

Enterprise Profile Clocks will interoperate with clocks operating in other profiles if the clocks in the other profiles obey the rules of the Enterprise Profile. These rules MUST NOT be changed to achieve interoperability with other profiles.

15. Security Considerations

Protocols used to transfer time, such as PTP and NTP can be important to security mechanisms which use time windows for keys and authorization. Passing time through the networks poses a security risk since time can potentially be manipulated. The use of multiple simultaneous masters, using multiple PTP domains can mitigate problems from rogue masters and man-in-the-middle attacks. See sections 9 and 10. Additional security mechanisms are outside the scope of this document.

16. IANA Considerations

There are no IANA requirements in this specification.

17. References

17.1. Normative References

[IEEE1588] IEEE std. 1588-2008, "IEEE Standard for a Precision Clock Synchronization for Networked Measurement and Control Systems." July, 2008.

[RFC768] Postel, J., "User Datagram Protocol," RFC 768, August, 980.

[RFC791] "Internet Protocol DARPA Internet Program Protocol Specification," RFC 791, September, 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S., Hinden, R., "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, December, 1998.

17.2. Informative References

[G8271] ITU-T G.8271/Y.1366, "Time and Phase Synchronization Aspects of Packet Networks" February, 2012.

[NTP] Mills, D., Martin, J., Burbank, J., Kasch, W., "Network Time Protocol Version 4: Protocol and Algorithms Specification," RFC 5905, June 2010.

18. Acknowledgments

The authors would like to thank members of IETF for reviewing and providing feedback on this draft.

This document was initially prepared using 2-Word-v2.0.template.dot.

19. Authors' Addresses

Doug Arnold
Meinberg USA
929 Salem End Road
Framingham, MA 01702
USA

Email: doug.arnold@meinberg-usa.com

Heiko Gerstung
Meinberg Funkuhren GmbH & Co. KG
Lange Wand 9
D-31812 Bad Pyrmont
Germany

Email: Heiko.gerstung@meinberg.de

TICTOC Working Group
INTERNET DRAFT
Intended status: Standards Track

Vinay Shankarkumar
Laurent Montini
Cisco Systems

Tim Frost
Calnex Solutions Ltd.

Greg Dowd
Microsemi

Expires: September 17, 2017

March 17, 2017

Precision Time Protocol Version 2 (PTPv2) Management Information Base
draft-ietf-tictoc-ptp-mib-12.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on March 17, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents

(http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it defines objects for managing networks using Precision Time Protocol (PTP), specified in IEEE Std. 1588(TM)-2008.

This memo specifies a MIB module in a manner that is both compliant to the SMIv2, and semantically identical to the peer SMIv1 definitions.

Table of Contents

1. Introduction 2
   1.1. Relationship to other Profiles and MIBs 3
   1.2. Change Log 3
2. The SNMP Management Framework 5
3. Overview 6
4. IETF PTP MIB Definition 6
5. Security Considerations 58
6. IANA Considerations 61
7. References 61
   7.1. Normative References 61
   7.2. Informative References 61
8. Acknowledgements 63
9. Author’s Addresses 63

1. Introduction

This memo defines a portion of the Management Information Base (MIB) module for use with network management protocols in the Internet Community. In particular, it describes managed objects used for managing PTP devices including the ordinary clock, transparent clock, boundary clocks.

This MIB module is restricted to reading standard PTP data elements, as described in [IEEE 1588-2008]. This enables it to monitor the operation of PTP clocks within the network. It is envisioned this MIB module will complement other managed objects to be defined that will provide more detailed information on the performance of PTP

clocks supporting the Telecom Profile defined in [G.8265.1], and any future profiles that may be defined. Those objects are considered out of scope for the current draft.

Similarly, this MIB module is read-only and not intended to provide the ability to configure PTP clocks. Since PTP clocks are often embedded in other network elements such as routers, switches and gateways, this ability is generally provided via the configuration interface for the network element.

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].

1.1. Relationship to other Profiles and MIBs

This MIB module is intended to be used with the default PTP profile described in [IEEE 1588-2008] when running over the IP network layer. As stated above, it is envisioned this MIB module will complement other managed objects to be defined to monitor and measure the performance of PTP clocks supporting specific PTP profiles, e.g. the Telecom Profile defined in [G.8265.1].

Some other PTP profiles have their own MIB modules defined as part of the profile, and this MIB module is not intended to replace those MIB modules.

1.2. Change Log

This section tracks changes made to the revisions of the Internet Drafts of this document. It will be *deleted* when the document is published as an RFC.

draft-vinay-tictoc-ptp-mib

-00 Mar 11 Initial version; showed structure of MIB

draft-ietf-tictoc-ptp-mib

-00 Jul 11 First full, syntactically correct and compileable MIB

-01 Jan 12 Revised following comments from Bert Wijnen:
- revised introduction to clarify the scope, and the relationship to other MIBs and profiles
- changed name to "ptpbase"
- corrected some data types
- corrected references and typos

-02 Jul 12 Revised following comment at IETF83:

- changed "ptpbaseClockPortRunningIPversion" to the more generic "ptpbaseClockPortRunningTransport", covering all transport types defined in [IEEE 1588-2008] (i.e. IPv4, IPv6, Ethernet, DeviceNet and ControlNet).
- changed addresses associated with transports from "InetAddress" (for the IP transport) to a string, to allow for the different transport types.

-03 Jul 12 Minor changes following comments from Andy Bierman:
- corrected some compilation errors
- moved OBJECT-GROUP and MODULE-COMPLIANCE macros to the end

-04 Jan 13 Changes:
- Use of 'AutonomousType' import
- Display hint being specified for ClockIdentity, ClockInterval, ClockPortTransportTypeAddress Textual Conventions
- Removal of the Textual convention ClockPortTransportType, replaced with the wellKnownTransportTypes
- Modified ptpbaseClockPortCurrentPeerAddressType, ptpbaseClockPortRunningTransport, ptpbaseClockPortAssociateAddressType, to use AutonomousType.
- various textual changes to descriptive text in response to comments

-05 Feb 13 Several changes in response to comments from Alun Luchuk and Kevin Gross:
- Modified the use of wellKnownTransportTypes and wellKnownEncapsulationTypes
- changed ptpbaseClockPortSyncOneStep to ptpbaseClockPortSyncTwoStep to match [IEEE 1588-2008] semantics
- Re-ordered textual conventions to be alphabetic
- Changed some types from Integer32 to use defined textual conventions
- various minor descriptive text changes

-06 Mar 14 Updated author information, and fixed typos

-07 Mar 15 Updated author information, and fixed typo/enum

-08 Feb 16 Updated MIB in response to Brian Haberman's comments:
- Fixed MIB date
- Fixed references to [IEEE 1588-2008]
- Changed "router" for "node"

-09 Apr 16 Updated following Dan Romascanu's MIB Doctor comments

-10 Aug 16 Update following further feedback from Dan Romascanu. Also updated security section to list out all objects with MAX-ACCESS other than non-accessible, in response to comments from Deborah Brungard and Alissa Cooper.

-11 Aug 16 Used corrected version of MIB text
- Reduced the DESCIPTION section and moved to section 3
- Added clarification that PtpClockIdentity can also be non-EUI-64 address
- Clarifications on PtpClockPortTransportTypeAddress, and mentioned counters being discontinuous
- Made PtpClockQualityClassType as enumeration
Updated overview section with a longer description.

-12 Mar 17 Replaced direct quotations of [IEEE 1588-2008] with references to avoid copyright issues.

2. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

o An overall architecture, described in STD62, [RFC 3411].

o Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16: [RFC 1155], [RFC 1212] and [RFC 1215].
The second version, called SMIv2, is described in STD 58: [RFC 2578], [RFC 2579] and [RFC 2580].

o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15 [RFC 1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in [RFC 1901] and [RFC 1906]. The third version of the message protocol is called SNMPv3 and described in STD62: [RFC 3417], [RFC 3412] and [RFC 3414].

o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15 [RFC 1157]. A second set of protocol operations and associated PDU formats is described in STD 62 [RFC 3416].

o A set of fundamental applications described in STD 62 [RFC 3413]

and the view-based access control mechanism described in STD 62 [RFC 3415].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIv2. A MIB module conforming to the SMIv1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (e.g., use of Counter64). Some machine readable information in SMIv2 will be converted into textual descriptions in SMIv1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB module.

3. Overview

The objects defined in this MIB module are to be used when describing the Precision Time Protocol (PTP), as defined in [IEEE 1588-2008].

Section 6 of [IEEE 1588-2008] provides an overview of synchronization networks using PTP.

Terms used in this document have meanings as defined in section 3.1 of [IEEE 1588-2008].

4. IETF PTP MIB Definition

```
PTPBASE-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    OBJECT-IDENTITY,
    Gauge32,
    Unsigned32,
    Counter32,
    Counter64,
    mib-2,
    Integer32
        FROM SNMPv2-SMI
    OBJECT-GROUP,
    MODULE-COMPLIANCE
        FROM SNMPv2-CONF
    TEXTUAL-CONVENTION,
    TruthValue,
    DisplayString,
```

```
    AutonomousType
        FROM SNMPv2-TC
    InterfaceIndexOrZero
        FROM IF-MIB;


ptpbaseMIB MODULE-IDENTITY
    LAST-UPDATED    "201703120000Z"
    ORGANIZATION    "TICTOC Working Group"
    CONTACT-INFO
          "WG Email: tictoc@ietf.org

           Vinay Shankarkumar
           Cisco Systems,
           Email: vinays@cisco.com

           Laurent Montini,
           Cisco Systems,
           Email: lmontini@cisco.com

           Tim Frost,
           Calnex Solutions Ltd.,
           Email: tim.frost@calnexsol.com

           Greg Dowd,
           Microsemi Inc.,
           Email: greg.dowd@microsemi.com"

    DESCRIPTION
        "The MIB module for PTP version 2 (IEEE Std. 1588(TM)-2008)

        Overview of PTP version 2 (IEEE Std. 1588(TM)-2008)

        [IEEE 1588-2008] defines a protocol enabling precise
        synchronization of clocks in measurement and control systems
        implemented with packet-based networks, the Precision Time
        Protocol Version 2 (PTPv2).  This MIB module does not address
        the earlier version IEEE Std. 1588(TM)-2002 (PTPv1).  The
        protocol is applicable to network elements communicating using
        IP.  The protocol enables heterogeneous systems that include
        clocks of various inherent precision, resolution, and stability
        to synchronize to a grandmaster clock.

        The protocol supports system-wide synchronization accuracy in
        the sub-microsecond range with minimal network and local clock
        computing resources. [IEEE 1588-2008] uses UDP/IP or
        Ethernet and can be adapted to other mappings. It includes
        formal mechanisms for message extensions, higher sampling rates,
        correction for asymmetry, a clock type to reduce error
```

accumulation in large topologies, and specifications on how to incorporate the resulting additional data into the synchronization protocol. The [IEEE 1588-2008] defines conformance and management capability also.

MIB description

This MIB module supports the Precision Time Protocol version 2 (PTPv2, hereafter designated as PTP) features of network element system devices, when using the default PTP profile described in [IEEE 1588-2008] when running over the IP network layer.

It is envisioned this MIB module will complement other managed objects to be defined to monitor and measure the performance of the PTP devices and telecom clocks supporting specific PTP profiles.

Some other PTP profiles have their own MIB modules defined as part of the profile, and this MIB module is not intended to replace those MIB modules.

Technical terms used in this module are defined in [IEEE 1588-2008].

The MIB module refers to the sections of [IEEE 1588-2008].

Acronyms:

ARB	Arbitrary Timescale
E2E	End-to-End
EUI	Extended Unique Identifier
GPS	Global Positioning System
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol
MAC	Media Access Control according to [IEEE 802.3-2008]
MAC-48	Used to identify hardware instances within 802-based networking applications. This is obsolete now.
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol (see IETF [RFC 5905])
OUI	Organizational Unique Identifier (allocated by the IEEE)
P2P	Peer-to-Peer
PTP	Precision Time Protocol
TAI	International Atomic Time
TC	Transparent Clock
UDP	User Datagram Protocol
UTC	Coordinated Universal Time

References:

[IEEE 1588-2008] IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, IEEE Std. 1588(TM)-2008, 24 July 2008.

The below table specifies the object formats of the various textual conventions used.

Data type mapping	Textual Convention	SYNTAX
5.3.2 TimeInterval	PtpClockTimeInterval	OCTET STRING(SIZE(1..255))
5.3.3 Timestamp	PtpClockTimestamp	OCTET STRING(SIZE(6))
5.3.4 ClockIdentity	PtpClockIdentity	OCTET STRING(SIZE(8))
5.3.5 PortIdentity	PtpClockPortNumber	INTEGER(1..65535)
5.3.7 ClockQuality	PtpClockQualityClassType	

"

```
      -- revision log
      REVISION        "201703120000Z"
      DESCRIPTION     "Draft 12, for IESG approval remvoed the IEEE
standard texts."

      REVISION        "201608240000Z"
      DESCRIPTION     "Draft 11, for IESG approval after all comments,
including the correct MIB."

      REVISION        "201608220000Z"
      DESCRIPTION     "Draft 10, for IESG approval after all comments
addressed."

      REVISION        "201604200000Z"
      DESCRIPTION     "Draft 9, for IESG approval."

      REVISION        "201602220000Z"
      DESCRIPTION     "Draft 8, for IETF last call."

      ::= { mib-2 XXX } -- XXX to be assigned by IANA


-- Textual Conventions

PtpClockDomainType ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "d"
    STATUS          current
    DESCRIPTION
        "The Domain is identified by an integer, the domainNumber, in
        the range of 0 to 255. An integer value that is used to assign
        each PTP device to a particular domain."
```

```
    REFERENCE   "Section 7.1 Domains, Table 2 of [IEEE 1588-2008]"
    SYNTAX      Unsigned32 (0..255)

PtpClockIdentity ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "255a"
    STATUS          current
    DESCRIPTION
        "The clock Identity is an 8-octet array and will be presented in
        the form of a character array.  Network byte order is assumed.

        The value of the PtpClockIdentity should be taken from the
        IEEE EUI-64 individual assigned numbers as indicated in
        Section 7.5.2.2.2 of [IEEE 1588-2008]. It can also be non-EUI-64
        address as defined in section 7.5.2.2.3 of [IEEE 1588-2008].

        The clock identifier can be constructed from existing EUI-48
        assignments and here is an abbreviated example extracted from
        section 7.5.2.2.2 [IEEE 1588-2008]."

    REFERENCE       "Section 7.5.2.2.1 of [IEEE 1588-2008]"
    SYNTAX          OCTET STRING (SIZE (8))

PtpClockInstanceType ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "d"
    STATUS          current
    DESCRIPTION
        "The instance of the Clock of a given clock type in a given
        domain."
    SYNTAX          Unsigned32 (0..255)

PtpClockIntervalBase2 ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "d"
    STATUS          current
    DESCRIPTION
        "The interval included in message types Announce, Sync,
        Delay_Req, and Pdelay_Req as indicated in section 7.7.2.1 of
        [IEEE 1588-2008]."

    REFERENCE   "Section 7.7.2.1 General interval specification of
                 [IEEE 1588-2008]"
    SYNTAX      Integer32 (-128..127)

PtpClockMechanismType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The clock type based on whether end-to-end or peer-to-peer
        mechanisms are used. The mechanism used to calculate the Mean
        Path Delay as indicated in Table 9 of [IEEE 1588-2008]."
```

```
    REFERENCE
        "Sections 8.2.5.4.4 portDS.delayMechanism,
        6.6.4  Measuring link propagation delay in clocks supporting
        peer-to-peer path correction,
        7.4.2 communication Path asymmetry of [IEEE 1588-2008]."
    SYNTAX      INTEGER {
                    e2e(1),
                    p2p(2),
                    disabled(254)
                    }

PtpClockPortNumber ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "d"
    STATUS          current
    DESCRIPTION
        "An index identifying a specific Precision Time Protocol (PTP)
        port on a PTP node."

    REFERENCE
        "Sections 7.5.2.3 portNumber and 5.3.5 PortIdentity of
        [IEEE 1588-2008]"
    SYNTAX      Unsigned32 (0..65535)

PtpClockPortState ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This is the value of the current state of the protocol engine
        associated with this port."

    REFERENCE
        "Section 8.2.5.3.1 portState and 9.2.5 State machines of
        [IEEE 1588-2008]"
    SYNTAX          INTEGER  {
                        initializing(1),
                        faulty(2),
                        disabled(3),
                        listening(4),
                        preMaster(5),
                        master(6),
                        passive(7),
                        uncalibrated(8),
                        slave(9)
                    }

PtpClockPortTransportTypeAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "255a"
    STATUS          current
    DESCRIPTION
```

```
        "The Clock port transport protocol address used for this
         communication between the clock nodes. This is a string
         corresponding to the address type as specified by the
         transport type used. The transport types can be defined
         elsewhere, in addition to the ones defined in this document.
         This can be an address of type IP version 4, IP version 6,
         Ethernet, DeviceNET, ControlNET or IEC61158. The OCTET STRING
         representation of the OID of ptpbaseWellKnownTransportTypes
         will be used in the values contained in the OCTET STRING."

    REFERENCE   "Annex D (IPv4), Annex E (IPv6), Annex F (Ethernet),
                 Annex G (DeviceNET), Annex H (ControlNET) and
                 Annex I (IEC61158) of [IEEE 1588-2008]"
    SYNTAX          OCTET STRING (SIZE (1..255))

PtpClockProfileType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "Clock Profile used. A profile is the set of allowed Precision
        Time Protocol (PTP) features applicable to a device."

    REFERENCE       "Section 3.1.30 profile and 19.3 PTP profiles of
                     [IEEE 1588-2008]"
    SYNTAX          INTEGER  {
                        default(1),
                        telecom(2),
                        vendorspecific(3)
                    }

PtpClockQualityAccuracyType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The ClockQuality as specified in sections 5.3.7, 7.6.2.5 and
        Table 6 of [IEEE 1588-2008].

        The following values are not represented in the enumerated
        values.

                 0x01-0x1F Reserved
                 0x32-0x7F Reserved

        It is important to note that section 7.1.1 of [RFC 2578] allows
        for gaps and enumerate values starting at zero when indicated by
        the protocol."

    REFERENCE
        "Section 5.3.7 ClockQuality, 7.6.2.5 clockAccuracy and Table 6
        clockAccuracy enumeration of [IEEE 1588-2008]"
    SYNTAX          INTEGER  {
```

```
                     -- reserved00(0:31), 0x00 to 0x1F
                        nanoSecond25(32),    -- 0x20
                        nanoSecond100(33),   -- 0x21
                        nanoSecond250(34),   -- 0x22
                        microSec1(35),       -- 0x23
                        microSec2dot5(36),   -- 0x24
                        microSec10(37),      -- 0x25
                        microSec25(38),      -- 0x26
                        microSec100(39),     -- 0x27
                        microSec250(40),     -- 0x28
                        milliSec1(41),       -- 0x29
                        milliSec2dot5(42),   -- 0x2A
                        milliSec10(43),      -- 0x2B
                        milliSec25(44),      -- 0x2C
                        milliSec100(45),     -- 0x2D
                        milliSec250(46),     -- 0x2E
                        second1(47),         -- 0x2F
                        second10(48),        -- 0x30
                        secondGreater10(49), -- 0x31
                        unknown(254)         -- 0xFE
                     -- reserved255(255),    0xFF
                    }

PtpClockQualityClassType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The ClockQuality as specified in section 5.3.7 ClockQuality,
        7.6.2.4 clockClass and Table 5 clockClass specifications of
        [IEEE 1588-2008]."

    REFERENCE       "Section 5.3.7, 7.6.2.4 and Table 5 of
                     [IEEE 1588-2008]."
    SYNTAX          INTEGER  {
                     -- reserved(0), 0x00
                     -- reserved(1:5), 0x01 to 0x05
                        clockclass6(6), -- 0x06
                        clockclass7(7), -- 0x07
                        -- reserved(8), 0x08
                        -- reserved(9:10), 0x09 to 0x0A
                        -- reserved(11:12), 0x0B, 0x0C
                        clockclass13(13), -- 0x0D
                        clockclass14(14), -- 0x0E
                        -- reserved(15:51), 0x0F to 0x33
                        clockclass52(52), -- 0x34
                        -- reserved(53:57), 0x35 to 0x39
                        clockclass58(58) -- 0x3A
                        -- reserved(59:67), 0x3B to 0x43
                        -- otherprofiles(68:122), 0x44 to 0x7A
                        -- reserved(123:127), 0x7B to 0x7F
```

```
                    -- reserved(128:132), 0x80 to 0x84
                }

PtpClockRoleType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The Clock Role. The protocol generates a Master Slave
        relationship among the clocks in the system.

        Clock Role      Value
        -------------------------
        Master clock     1
        Slave clock      2      "
    SYNTAX          INTEGER  {
                        master(1),
                        slave(2)
                    }

PtpClockStateType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The clock state returned by a PTP engine.

        Clock State      Value
        ------------------------
        Freerun state       1
        Holdover state      2
        Acquiring state     3
        Freq_locked state   4
        Phase_aligned state 5  "
    SYNTAX          INTEGER  {
                        freerun(1),
                        holdover(2),
                        acquiring(3),
                        frequencyLocked(4),
                        phaseAligned(5)
                    }

PtpClockTimeInterval ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    "255a"
    STATUS          current
    DESCRIPTION
        "This textual convention corresponds to the TimeInterval
        structure indicated in section 5.3.2 of [IEEE 1588-2008].
        It will be presented in the form of a character array.
        Network byte order is assumed."

    REFERENCE
        "Section 5.3.2 TimeInterval and section 7.7.2.1 Timer interval
```

```
         specification of [IEEE 1588-2008]"
    SYNTAX          OCTET STRING (SIZE (1..255))

PtpClockTimeSourceType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "The ClockQuality as specified in Sections 5.3.7, 7.6.2.6 and
        Table 7 of [IEEE 1588-2008].

        The following values are not represented in the enumerated
        values.

        0xF0-0xFE  For use by alternate PTP profiles
        0xFF       Reserved

        It is important to note that section 7.1.1 RFC 2578 allows for
        gaps and enumerate values to start with zero when indicated by
        the protocol."

    REFERENCE       "Section 5.3.7, 7.6.2.6 and Table 7 of
                     [IEEE 1588-2008]."
    SYNTAX          INTEGER  {
                        atomicClock(16), -- 0x10
                        gps(32), -- 0x20
                        terrestrialRadio(48), -- 0x22
                        ptp(64), -- 0x40
                        ntp(80), -- 0x50
                        handSet(96), -- 0x60
                        other(144), -- 0x90
                        internalOscillator(160) -- 0xA0
                    }

PtpClockTxModeType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "Transmission mode.

        Unicast:       Using unicast communication channel.
        Multicast:     Using Multicast communication channel.
        multicast-mix: Using multicast-unicast communication channel"
    SYNTAX          INTEGER  {
                        unicast(1),
                        multicast(2),
                        multicastmix(3)
                    }

PtpClockType ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
```

```
        "The clock types as defined in the MIB module description."

    REFERENCE
        "Section 6.5.1 PTP device types of [IEEE 1588-2008]."
    SYNTAX          INTEGER  {
                        ordinaryClock(1),
                        boundaryClock(2),
                        transparentClock(3),
                        boundaryNode(4)
                    }

ptpbaseMIBNotifs  OBJECT IDENTIFIER
    ::= { ptpbaseMIB 0 }

ptpbaseMIBObjects  OBJECT IDENTIFIER
    ::= { ptpbaseMIB 1 }

ptpbaseMIBConformance  OBJECT IDENTIFIER
    ::= { ptpbaseMIB 2 }

ptpbaseMIBSystemInfo  OBJECT IDENTIFIER
    ::= { ptpbaseMIBObjects 1 }

ptpbaseMIBClockInfo  OBJECT IDENTIFIER
    ::= { ptpbaseMIBObjects 2 }


ptpbaseSystemTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseSystemEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of count information about the PTP system for all
        domains."
    ::= { ptpbaseMIBSystemInfo 1 }

ptpbaseSystemEntry OBJECT-TYPE
    SYNTAX          PtpbaseSystemEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing count information about a
        single domain. New row entries are added when the PTP clock for
        this domain is configured, while the unconfiguration of the PTP
        clock removes it."
    INDEX           {
                        ptpDomainIndex,
                        ptpInstanceIndex
                    }
```

```
    ::= { ptpbaseSystemTable 1 }

PtpbaseSystemEntry ::= SEQUENCE {
        ptpDomainIndex           PtpClockDomainType,
        ptpInstanceIndex         PtpClockInstanceType,
        ptpDomainClockPortsTotal Gauge32
}

ptpDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
        logical group of PTP devices. The Clock Domain is a logical
        group of clocks and devices that synchronize with each other
        using the PTP protocol.

        0           Default domain
        1           Alternate domain 1
        2           Alternate domain 2
        3           Alternate domain 3
        4 - 127     User-defined domains
        128 - 255   Reserved"
    ::= { ptpbaseSystemEntry 1 }

ptpInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the Clock for this
        domain."
    ::= { ptpbaseSystemEntry 2 }

ptpDomainClockPortsTotal OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "ptp ports"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the total number of clock ports
        configured within a domain in the system."
    ::= { ptpbaseSystemEntry 3 }



ptpbaseSystemDomainTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseSystemDomainEntry
```

```
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP system for all clock modes
        -- ordinary, boundary or transparent."
    ::= { ptpbaseMIBSystemInfo 2 }

ptpbaseSystemDomainEntry OBJECT-TYPE
    SYNTAX          PtpbaseSystemDomainEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        clock mode for the PTP system. A row entry gets added when PTP
        clocks are configured on the node."
    INDEX           { ptpbaseSystemDomainClockTypeIndex }
    ::= { ptpbaseSystemDomainTable 1 }

PtpbaseSystemDomainEntry ::= SEQUENCE {
        ptpbaseSystemDomainClockTypeIndex PtpClockType,
        ptpbaseSystemDomainTotals         Unsigned32
}

ptpbaseSystemDomainClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseSystemDomainEntry 1 }

ptpbaseSystemDomainTotals OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "domains"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the total number of PTP domains for this
        particular clock type configured in this node."
    ::= { ptpbaseSystemDomainEntry 2 }



ptpbaseSystemProfile OBJECT-TYPE
    SYNTAX          PtpClockProfileType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
```

```
        "This object specifies the PTP Profile implemented on the
        system."
    REFERENCE       "Section 19.3 PTP profiles of [IEEE 1588-2008]"
    ::= { ptpbaseMIBSystemInfo 3 }

ptpbaseClockCurrentDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockCurrentDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock Current Datasets for
        all domains."
    ::= { ptpbaseMIBClockInfo 1 }

ptpbaseClockCurrentDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockCurrentDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP clock Current Datasets for a domain."
    REFERENCE
        "[IEEE 1588-2008] Section 8.2.2 currentDS data set member
        specifications of [IEEE 1588-2008]"
    INDEX           {
                        ptpbaseClockCurrentDSDomainIndex,
                        ptpbaseClockCurrentDSClockTypeIndex,
                        ptpbaseClockCurrentDSInstanceIndex
                    }
    ::= { ptpbaseClockCurrentDSTable 1 }

PtpbaseClockCurrentDSEntry ::= SEQUENCE {
        ptpbaseClockCurrentDSDomainIndex      PtpClockDomainType,
        ptpbaseClockCurrentDSClockTypeIndex   PtpClockType,
        ptpbaseClockCurrentDSInstanceIndex    PtpClockInstanceType,
        ptpbaseClockCurrentDSStepsRemoved     Unsigned32,
        ptpbaseClockCurrentDSOffsetFromMaster PtpClockTimeInterval,
        ptpbaseClockCurrentDSMeanPathDelay    PtpClockTimeInterval
}

ptpbaseClockCurrentDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
logical
        group of PTP devices."
    ::= { ptpbaseClockCurrentDSEntry 1 }
```

```
ptpbaseClockCurrentDSClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockCurrentDSEntry 2 }

ptpbaseClockCurrentDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockCurrentDSEntry 3 }

ptpbaseClockCurrentDSStepsRemoved OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "Steps"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The current clock dataset StepsRemoved value.

        This object specifies the distance measured by the number of
        Boundary clocks between the local clock and the Foreign master
        as indicated in the stepsRemoved field of Announce messages."
    REFERENCE
        "Section 8.2.2.2 stepsRemoved of [IEEE 1588-2008]"
    ::= { ptpbaseClockCurrentDSEntry 4 }

ptpbaseClockCurrentDSOffsetFromMaster OBJECT-TYPE
    SYNTAX          PtpClockTimeInterval
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the current clock dataset ClockOffset
        value. The value of the computation of the offset in time
        between a slave and a master clock."
    REFERENCE
        "Section 8.2.2.3 currentDS.offsetFromMaster of [IEEE 1588-2008]"
    ::= { ptpbaseClockCurrentDSEntry 5 }

ptpbaseClockCurrentDSMeanPathDelay OBJECT-TYPE
    SYNTAX          PtpClockTimeInterval
```

```
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the current clock dataset
        MeanPathDelay value.

        The mean path delay between a pair of ports as measured by the
        delay request-response mechanism."
    REFERENCE
        "Section 8.2.2.4 currentDS.meanPathDelay of [IEEE 1588-2008]"
    ::= { ptpbaseClockCurrentDSEntry 6 }



ptpbaseClockParentDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockParentDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock Parent Datasets for
        all domains."
    ::= { ptpbaseMIBClockInfo 2 }

ptpbaseClockParentDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockParentDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP clock Parent Datasets for a domain."
    REFERENCE
        "Section 8.2.3 parentDS data set member specifications of
        [IEEE 1588-2008]"
    INDEX           {
                        ptpbaseClockParentDSDomainIndex,
                        ptpbaseClockParentDSClockTypeIndex,
                        ptpbaseClockParentDSInstanceIndex
                    }
    ::= { ptpbaseClockParentDSTable 1 }

PtpbaseClockParentDSEntry ::= SEQUENCE {
    ptpbaseClockParentDSDomainIndex           PtpClockDomainType,
    ptpbaseClockParentDSClockTypeIndex        PtpClockType,
    ptpbaseClockParentDSInstanceIndex         PtpClockInstanceType,
    ptpbaseClockParentDSParentPortIdentity    OCTET STRING,
    ptpbaseClockParentDSParentStats           TruthValue,
    ptpbaseClockParentDSOffset                PtpClockIntervalBase2,
    ptpbaseClockParentDSClockPhChRate         Integer32,
```

```
    ptpbaseClockParentDSGMClockIdentity        PtpClockIdentity,
    ptpbaseClockParentDSGMClockPriority1       Unsigned32,
    ptpbaseClockParentDSGMClockPriority2       Unsigned32,
    ptpbaseClockParentDSGMClockQualityClass    PtpClockQualityClassType,
    ptpbaseClockParentDSGMClockQualityAccuracy
PtpClockQualityAccuracyType,
    ptpbaseClockParentDSGMClockQualityOffset   Unsigned32
}

ptpbaseClockParentDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
logical
        group of PTP devices."
    ::= { ptpbaseClockParentDSEntry 1 }

ptpbaseClockParentDSClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockParentDSEntry 2 }

ptpbaseClockParentDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockParentDSEntry 3 }

ptpbaseClockParentDSParentPortIdentity OBJECT-TYPE
    SYNTAX          OCTET STRING(SIZE(1..256))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value of portIdentity of the port on
        the master that issues the Sync messages used in synchronizing
        this clock."
    REFERENCE
        "Section 8.2.3.2 parentDS.parentPortIdentity of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 4 }
```

```
ptpbaseClockParentDSParentStats OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Parent Dataset ParentStats value.

        This value indicates whether the values of ParentDSOffset
        and ParentDSClockPhChRate have been measured and are valid.
        A TRUE value shall indicate valid data."
    REFERENCE
        "Section 8.2.3.3 parentDS.parentStats of [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 5 }

ptpbaseClockParentDSOffset OBJECT-TYPE
    SYNTAX          PtpClockIntervalBase2 (-128..127)
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Parent Dataset
        ParentOffsetScaledLogVariance value.

        This value is the variance of the parent clock's phase as
        measured by the local clock."
    REFERENCE
        "Section 8.2.3.4
        parentDS.observedParentOffsetScaledLogVariance
        [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 6 }

ptpbaseClockParentDSClockPhChRate OBJECT-TYPE
    SYNTAX          Integer32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the clock's parent dataset
        ParentClockPhaseChangeRate value.

        This value is an estimate of the parent clock's phase change
        rate as measured by the slave clock."
    REFERENCE
        "Section 8.2.3.5
        parentDS.observedParentClockPhaseChangeRate of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 7 }

ptpbaseClockParentDSGMClockIdentity OBJECT-TYPE
    SYNTAX          PtpClockIdentity
```

```
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset Grandmaster clock
        identity."
    REFERENCE
        "Section 8.2.3.6 parentDS.grandmasterIdentity of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 8 }

ptpbaseClockParentDSGMClockPriority1 OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset Grandmaster clock
        priority1."
    REFERENCE
        "Section 8.2.3.8 parentDS.grandmasterPriority1 of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 9 }

ptpbaseClockParentDSGMClockPriority2 OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset grandmaster clock
        priority2."
    REFERENCE
        "Section 8.2.3.9 parentDS.grandmasterPriority2 of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 10 }

ptpbaseClockParentDSGMClockQualityClass OBJECT-TYPE
    SYNTAX          PtpClockQualityClassType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset grandmaster clock
        quality class."
    REFERENCE
        "Section 8.2.3.7 parentDS.grandmasterClockQuality of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 11 }

ptpbaseClockParentDSGMClockQualityAccuracy OBJECT-TYPE
    SYNTAX          PtpClockQualityAccuracyType
    MAX-ACCESS      read-only
```

```
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset grandmaster clock
        quality accuracy."
    REFERENCE
        "Section 8.2.3.7 parentDS.grandmasterClockQuality of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 12 }

ptpbaseClockParentDSGMClockQualityOffset OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the parent dataset grandmaster clock
        quality offset."
    REFERENCE
        "Section 8.2.3.7 parentDS.grandmasterClockQuality of
         [IEEE 1588-2008]"
    ::= { ptpbaseClockParentDSEntry 13 }



ptpbaseClockDefaultDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockDefaultDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock Default Datasets for
        all domains."
    ::= { ptpbaseMIBClockInfo 3 }

ptpbaseClockDefaultDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockDefaultDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP clock Default Datasets for a domain."
    INDEX           {
                        ptpbaseClockDefaultDSDomainIndex,
                        ptpbaseClockDefaultDSClockTypeIndex,
                        ptpbaseClockDefaultDSInstanceIndex
                    }
    ::= { ptpbaseClockDefaultDSTable 1 }

PtpbaseClockDefaultDSEntry ::= SEQUENCE {
        ptpbaseClockDefaultDSDomainIndex     PtpClockDomainType,
        ptpbaseClockDefaultDSClockTypeIndex  PtpClockType,
```

```
        ptpbaseClockDefaultDSInstanceIndex   PtpClockInstanceType,
        ptpbaseClockDefaultDSTwoStepFlag     TruthValue,
        ptpbaseClockDefaultDSClockIdentity   PtpClockIdentity,
        ptpbaseClockDefaultDSPriority1       Unsigned32,
        ptpbaseClockDefaultDSPriority2       Unsigned32,
        ptpbaseClockDefaultDSSlaveOnly       TruthValue,
        ptpbaseClockDefaultDSQualityClass    PtpClockQualityClassType,
        ptpbaseClockDefaultDSQualityAccuracy
PtpClockQualityAccuracyType,
        ptpbaseClockDefaultDSQualityOffset   Integer32
}

ptpbaseClockDefaultDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
logical
        group of PTP devices."
    ::= { ptpbaseClockDefaultDSEntry 1 }

ptpbaseClockDefaultDSClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockDefaultDSEntry 2 }

ptpbaseClockDefaultDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockDefaultDSEntry 3 }

ptpbaseClockDefaultDSTwoStepFlag OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies whether the Two Step process is used."
    ::= { ptpbaseClockDefaultDSEntry 4 }

ptpbaseClockDefaultDSClockIdentity OBJECT-TYPE
```

```
    SYNTAX          PtpClockIdentity
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default Datasets clock identity."
    ::= { ptpbaseClockDefaultDSEntry 5 }

ptpbaseClockDefaultDSPriority1 OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default Datasets clock Priority1."
    ::= { ptpbaseClockDefaultDSEntry 6 }

ptpbaseClockDefaultDSPriority2 OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default Datasets clock Priority2."
    ::= { ptpbaseClockDefaultDSEntry 7 }

ptpbaseClockDefaultDSSlaveOnly OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Whether the SlaveOnly flag is set."
    ::= { ptpbaseClockDefaultDSEntry 8 }

ptpbaseClockDefaultDSQualityClass OBJECT-TYPE
    SYNTAX          PtpClockQualityClassType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default dataset Quality Class."
    ::= { ptpbaseClockDefaultDSEntry 9 }

ptpbaseClockDefaultDSQualityAccuracy OBJECT-TYPE
    SYNTAX          PtpClockQualityAccuracyType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default dataset Quality Accuracy."
    ::= { ptpbaseClockDefaultDSEntry 10 }

ptpbaseClockDefaultDSQualityOffset OBJECT-TYPE
    SYNTAX          Integer32
```

```
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the default dataset Quality offset."
    ::= { ptpbaseClockDefaultDSEntry 11 }



ptpbaseClockRunningTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockRunningEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock Running Datasets for
        all domains."
    ::= { ptpbaseMIBClockInfo 4 }

ptpbaseClockRunningEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockRunningEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP clock running Datasets for a domain."
    INDEX           {
                        ptpbaseClockRunningDomainIndex,
                        ptpbaseClockRunningClockTypeIndex,
                        ptpbaseClockRunningInstanceIndex
                    }
    ::= { ptpbaseClockRunningTable 1 }

PtpbaseClockRunningEntry ::= SEQUENCE {
        ptpbaseClockRunningDomainIndex     PtpClockDomainType,
        ptpbaseClockRunningClockTypeIndex  PtpClockType,
        ptpbaseClockRunningInstanceIndex   PtpClockInstanceType,
        ptpbaseClockRunningState           PtpClockStateType,
        ptpbaseClockRunningPacketsSent     Counter64,
        ptpbaseClockRunningPacketsReceived Counter64
}

ptpbaseClockRunningDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
        Logical group of PTP devices."
    ::= { ptpbaseClockRunningEntry 1 }
```

```
ptpbaseClockRunningClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockRunningEntry 2 }

ptpbaseClockRunningInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockRunningEntry 3 }

ptpbaseClockRunningState OBJECT-TYPE
    SYNTAX          PtpClockStateType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Clock state returned by a PTP
engine."

    ::= { ptpbaseClockRunningEntry 4 }

ptpbaseClockRunningPacketsSent OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the total number of all unicast and
        multicast packets that have been sent out for this clock in this
        domain for this type. These counters are discontinuous."
    ::= { ptpbaseClockRunningEntry 5 }

ptpbaseClockRunningPacketsReceived OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the total number of all unicast and
        multicast packets that have been received for this clock in this
        domain for this type. These counters are discontinuous."
    ::= { ptpbaseClockRunningEntry 6 }
```

```
ptpbaseClockTimePropertiesDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockTimePropertiesDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock time properties
        datasets for all domains."
    ::= { ptpbaseMIBClockInfo 5 }

ptpbaseClockTimePropertiesDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockTimePropertiesDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP clock timeproperties Datasets for a domain."
    REFERENCE
        "Section 8.2.4 timePropertiesDS data set member specifications
        of [IEEE 1588-2008]"
    INDEX           {
                        ptpbaseClockTimePropertiesDSDomainIndex,
                        ptpbaseClockTimePropertiesDSClockTypeIndex,
                        ptpbaseClockTimePropertiesDSInstanceIndex
                    }
    ::= { ptpbaseClockTimePropertiesDSTable 1 }

PtpbaseClockTimePropertiesDSEntry ::= SEQUENCE {
  ptpbaseClockTimePropertiesDSDomainIndex           PtpClockDomainType,
  ptpbaseClockTimePropertiesDSClockTypeIndex        PtpClockType,
  ptpbaseClockTimePropertiesDSInstanceIndex
PtpClockInstanceType,
  ptpbaseClockTimePropertiesDSCurrentUTCOffsetValid TruthValue,
  ptpbaseClockTimePropertiesDSCurrentUTCOffset      Integer32,
  ptpbaseClockTimePropertiesDSLeap59                TruthValue,
  ptpbaseClockTimePropertiesDSLeap61                TruthValue,
  ptpbaseClockTimePropertiesDSTimeTraceable         TruthValue,
  ptpbaseClockTimePropertiesDSFreqTraceable         TruthValue,
  ptpbaseClockTimePropertiesDSPTPTimescale          TruthValue,
  ptpbaseClockTimePropertiesDSSource
PtpClockTimeSourceType
}

ptpbaseClockTimePropertiesDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
```

```
        "This object specifies the domain number used to create a
logical
        group of PTP devices."
    ::= { ptpbaseClockTimePropertiesDSEntry 1 }

ptpbaseClockTimePropertiesDSClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockTimePropertiesDSEntry 2 }

ptpbaseClockTimePropertiesDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockTimePropertiesDSEntry 3 }

ptpbaseClockTimePropertiesDSCurrentUTCOffsetValid OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the timeproperties dataset value of
        whether the current UTC offset is valid."
    REFERENCE
        "Section 8.2.4.2 timePropertiesDS.currentUtcOffset of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 4 }

ptpbaseClockTimePropertiesDSCurrentUTCOffset OBJECT-TYPE
    SYNTAX          Integer32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the timeproperties dataset value of
        the current UTC offset.

        In PTP systems whose epoch is the PTP epoch, the value of
        timePropertiesDS.currentUtcOffset is the offset
        between TAI and UTC; otherwise the value has no meaning. The
        value shall be in units of seconds."
    REFERENCE
        "Section 8.2.4.3 timePropertiesDS.currentUtcOffsetValid of
```

```
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 5 }

ptpbaseClockTimePropertiesDSLeap59 OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Leap59 value in the clock Current
        Dataset."
    REFERENCE
        "Section 8.2.4.4 timePropertiesDS.leap59 of [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 6 }

ptpbaseClockTimePropertiesDSLeap61 OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Leap61 value in the clock Current
        Dataset."
    REFERENCE
        "Section 8.2.4.5 timePropertiesDS.leap61 of [IEEE 1588-2008]"
 ::= { ptpbaseClockTimePropertiesDSEntry 7 }

ptpbaseClockTimePropertiesDSTimeTraceable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Time Traceable value in the clock
        Current Dataset."
    REFERENCE
        "Section 8.2.4.6 timePropertiesDS.timeTraceable of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 8 }

ptpbaseClockTimePropertiesDSFreqTraceable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Frequency Traceable value in the
        clock Current Dataset."
    REFERENCE
        "Section 8.2.4.7 timePropertiesDS.frequencyTraceable of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 9 }
```

```
ptpbaseClockTimePropertiesDSPTPTimescale OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP Timescale value in the clock
        Current Dataset."
    REFERENCE
        "Section 8.2.4.8 timePropertiesDS.ptpTimescale of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 10 }

ptpbaseClockTimePropertiesDSSource OBJECT-TYPE
    SYNTAX          PtpClockTimeSourceType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Timesource value in the clock Current
        Dataset."
    REFERENCE
        "Section 8.2.4.9 timePropertiesDS.timeSource of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTimePropertiesDSEntry 11 }



ptpbaseClockTransDefaultDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockTransDefaultDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP Transparent clock Default
        Datasets for all domains."
    ::= { ptpbaseMIBClockInfo 6 }

ptpbaseClockTransDefaultDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockTransDefaultDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        PTP Transparent clock Default Datasets for a domain."
    REFERENCE
        "Section 8.3.2 transparentClockDefaultDS data set member
        specifications of [IEEE 1588-2008]"
    INDEX           {
                        ptpbaseClockTransDefaultDSDomainIndex,
                        ptpbaseClockTransDefaultDSInstanceIndex
                    }
```

```
    ::= { ptpbaseClockTransDefaultDSTable 1 }

PtpbaseClockTransDefaultDSEntry ::= SEQUENCE {
        ptpbaseClockTransDefaultDSDomainIndex   PtpClockDomainType,
        ptpbaseClockTransDefaultDSInstanceIndex PtpClockInstanceType,
        ptpbaseClockTransDefaultDSClockIdentity PtpClockIdentity,
        ptpbaseClockTransDefaultDSNumOfPorts    Counter32,
        ptpbaseClockTransDefaultDSDelay         PtpClockMechanismType,
        ptpbaseClockTransDefaultDSPrimaryDomain PtpClockDomainType
}

ptpbaseClockTransDefaultDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
logical
        group of PTP devices."
    ::= { ptpbaseClockTransDefaultDSEntry 1 }

ptpbaseClockTransDefaultDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockTransDefaultDSEntry 2 }

ptpbaseClockTransDefaultDSClockIdentity OBJECT-TYPE
    SYNTAX          PtpClockIdentity
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value of the clockIdentity attribute
        of the local clock."
    REFERENCE
       "Section 8.3.2.2.1 transparentClockDefaultDS.clockIdentity of
       [IEEE 1588-2008]"
    ::= { ptpbaseClockTransDefaultDSEntry 3 }

ptpbaseClockTransDefaultDSNumOfPorts OBJECT-TYPE
    SYNTAX          Counter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the number of PTP ports of the device.
         These counters are discontinuous."
```

```
    REFERENCE
        "Section 8.3.2.2.2 transparentClockDefaultDS.numberPorts of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTransDefaultDSEntry 4 }

ptpbaseClockTransDefaultDSDelay OBJECT-TYPE
    SYNTAX          PtpClockMechanismType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object, if the transparent clock is an end-to-end
        transparent clock, has the value of E2E; if the
        transparent clock is a peer-to-peer transparent clock, the
        value
        shall be P2P."
    REFERENCE
        "Section 8.3.2.3.1 transparentClockDefaultDS.delayMechanism of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTransDefaultDSEntry 5 }

ptpbaseClockTransDefaultDSPrimaryDomain OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value of the primary syntonization
        domain. The initialization value shall be 0."
    REFERENCE
        "Section 8.3.2.3.2 transparentClockDefaultDS.primaryDomain of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockTransDefaultDSEntry 6 }



ptpbaseClockPortTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockPortEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the clock ports for a particular
        domain."
    ::= { ptpbaseMIBClockInfo 7 }

ptpbaseClockPortEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockPortEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
```

```
        clock port."
    INDEX           {
                        ptpbaseClockPortDomainIndex,
                        ptpbaseClockPortClockTypeIndex,
                        ptpbaseClockPortClockInstanceIndex,
                        ptpbaseClockPortTablePortNumberIndex
                    }
    ::= { ptpbaseClockPortTable 1 }

PtpbaseClockPortEntry ::= SEQUENCE {
        ptpbaseClockPortDomainIndex            PtpClockDomainType,
        ptpbaseClockPortClockTypeIndex         PtpClockType,
        ptpbaseClockPortClockInstanceIndex     PtpClockInstanceType,
        ptpbaseClockPortTablePortNumberIndex   PtpClockPortNumber,
        ptpbaseClockPortName                   DisplayString,
        ptpbaseClockPortRole                   PtpClockRoleType,
        ptpbaseClockPortSyncTwoStep            TruthValue,
        ptpbaseClockPortCurrentPeerAddressType AutonomousType,
        ptpbaseClockPortCurrentPeerAddress
PtpClockPortTransportTypeAddress,
        ptpbaseClockPortNumOfAssociatedPorts   Gauge32
}

ptpbaseClockPortDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
        logical group of PTP devices."
    ::= { ptpbaseClockPortEntry 1 }

ptpbaseClockPortClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockPortEntry 2 }

ptpbaseClockPortClockInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockPortEntry 3 }
```

```
ptpbaseClockPortTablePortNumberIndex OBJECT-TYPE
    SYNTAX          PtpClockPortNumber
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP Portnumber for this port."
    ::= { ptpbaseClockPortEntry 4 }

ptpbaseClockPortName OBJECT-TYPE
    SYNTAX          DisplayString (SIZE  (1..64))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP clock port name configured on the
        node."
    ::= { ptpbaseClockPortEntry 5 }

ptpbaseClockPortRole OBJECT-TYPE
    SYNTAX          PtpClockRoleType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object describes the current role (slave/master) of the
        port."
    ::= { ptpbaseClockPortEntry 6 }

ptpbaseClockPortSyncTwoStep OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies that two-step clock operation between
        the PTP master and slave device is enabled."
    ::= { ptpbaseClockPortEntry 7 }

ptpbaseClockPortCurrentPeerAddressType OBJECT-TYPE
    SYNTAX          AutonomousType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the current peer's network address type
         used for PTP communication."
    ::= { ptpbaseClockPortEntry 8 }

ptpbaseClockPortCurrentPeerAddress OBJECT-TYPE
    SYNTAX          PtpClockPortTransportTypeAddress
    MAX-ACCESS      read-only
    STATUS          current
```

```
    DESCRIPTION
        "This object specifies the current peer's network address used
        for PTP communication."
    ::= { ptpbaseClockPortEntry 9 }

ptpbaseClockPortNumOfAssociatedPorts OBJECT-TYPE
    SYNTAX          Gauge32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies -
        For a master port - the number of PTP slave sessions (peers)
        associated with this PTP port.
        For a slave port - the number of masters available to this slave
        port (might or might not be peered)."
    ::= { ptpbaseClockPortEntry 10 }



ptpbaseClockPortDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockPortDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the clock ports dataset for a
        particular domain."
    ::= { ptpbaseMIBClockInfo 8 }

ptpbaseClockPortDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockPortDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing port dataset information for
        a single clock port."
    INDEX           {
                        ptpbaseClockPortDSDomainIndex,
                        ptpbaseClockPortDSClockTypeIndex,
                        ptpbaseClockPortDSClockInstanceIndex,
                        ptpbaseClockPortDSPortNumberIndex
                    }
    ::= { ptpbaseClockPortDSTable 1 }

PtpbaseClockPortDSEntry ::= SEQUENCE {
        ptpbaseClockPortDSDomainIndex          PtpClockDomainType,
        ptpbaseClockPortDSClockTypeIndex       PtpClockType,
        ptpbaseClockPortDSClockInstanceIndex   PtpClockInstanceType,
        ptpbaseClockPortDSPortNumberIndex      PtpClockPortNumber,
        ptpbaseClockPortDSName                 DisplayString,
```

```
        ptpbaseClockPortDSPortIdentity         OCTET STRING,
        ptpbaseClockPortDSlogAnnouncementInterval PtpClockIntervalBase2,
        ptpbaseClockPortDSAnnounceRctTimeout   Integer32,
        ptpbaseClockPortDSlogSyncInterval      PtpClockIntervalBase2,
        ptpbaseClockPortDSMinDelayReqInterval  Integer32,
        ptpbaseClockPortDSPeerDelayReqInterval Integer32,
        ptpbaseClockPortDSDelayMech            PtpClockMechanismType,
        ptpbaseClockPortDSPeerMeanPathDelay    PtpClockTimeInterval,
        ptpbaseClockPortDSGrantDuration        Unsigned32,
        ptpbaseClockPortDSPTPVersion           Unsigned32
}

ptpbaseClockPortDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
        logical group of PTP devices."
    ::= { ptpbaseClockPortDSEntry 1 }

ptpbaseClockPortDSClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockPortDSEntry 2 }

ptpbaseClockPortDSClockInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockPortDSEntry 3 }

ptpbaseClockPortDSPortNumberIndex OBJECT-TYPE
    SYNTAX          PtpClockPortNumber
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP portnumber associated with this
        PTP port."
    ::= { ptpbaseClockPortDSEntry 4 }

ptpbaseClockPortDSName OBJECT-TYPE
```

```
    SYNTAX          DisplayString (SIZE  (1..64))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP clock port dataset name."
    ::= { ptpbaseClockPortDSEntry 5 }

ptpbaseClockPortDSPortIdentity OBJECT-TYPE
    SYNTAX          OCTET STRING(SIZE(1..256))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP clock port Identity."
    ::= { ptpbaseClockPortDSEntry 6 }

ptpbaseClockPortDSlogAnnouncementInterval OBJECT-TYPE
    SYNTAX          PtpClockIntervalBase2
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Announce message transmission
        interval associated with this clock port."
    ::= { ptpbaseClockPortDSEntry 7 }

ptpbaseClockPortDSAnnounceRctTimeout OBJECT-TYPE
    SYNTAX          Integer32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Announce receipt timeout associated
        with this clock port."
    ::= { ptpbaseClockPortDSEntry 8 }

ptpbaseClockPortDSlogSyncInterval OBJECT-TYPE
    SYNTAX          PtpClockIntervalBase2
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Sync message transmission interval."
    ::= { ptpbaseClockPortDSEntry 9 }

ptpbaseClockPortDSMinDelayReqInterval OBJECT-TYPE
    SYNTAX          Integer32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Delay_Req message transmission
```

```
        interval."
    ::= { ptpbaseClockPortDSEntry 10 }

ptpbaseClockPortDSPeerDelayReqInterval OBJECT-TYPE
    SYNTAX          Integer32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Pdelay_Req message transmission
        interval."
    ::= { ptpbaseClockPortDSEntry 11 }

ptpbaseClockPortDSDelayMech OBJECT-TYPE
    SYNTAX          PtpClockMechanismType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the delay mechanism used. If the clock
        is an end-to-end clock, the value of the is e2e, else if the
        clock is a peer to-peer clock, the value shall be p2p."
    ::= { ptpbaseClockPortDSEntry 12 }

ptpbaseClockPortDSPeerMeanPathDelay OBJECT-TYPE
    SYNTAX          PtpClockTimeInterval
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the peer meanPathDelay."
    ::= { ptpbaseClockPortDSEntry 13 }

ptpbaseClockPortDSGrantDuration OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "seconds"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the grant duration allocated by the
        master."
    ::= { ptpbaseClockPortDSEntry 14 }

ptpbaseClockPortDSPTPVersion OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP version being used."
    ::= { ptpbaseClockPortDSEntry 15 }
```

```
ptpbaseClockPortRunningTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockPortRunningEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the clock ports running datasets for
        a particular domain."
    ::= { ptpbaseMIBClockInfo 9 }

ptpbaseClockPortRunningEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockPortRunningEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing running dataset information
        about a single clock port."
    INDEX           {
                        ptpbaseClockPortRunningDomainIndex,
                        ptpbaseClockPortRunningClockTypeIndex,
                        ptpbaseClockPortRunningClockInstanceIndex,
                        ptpbaseClockPortRunningPortNumberIndex
                    }
    ::= { ptpbaseClockPortRunningTable 1 }

PtpbaseClockPortRunningEntry ::= SEQUENCE {
        ptpbaseClockPortRunningDomainIndex        PtpClockDomainType,
        ptpbaseClockPortRunningClockTypeIndex     PtpClockType,
        ptpbaseClockPortRunningClockInstanceIndex PtpClockInstanceType,
        ptpbaseClockPortRunningPortNumberIndex    PtpClockPortNumber,
        ptpbaseClockPortRunningName               DisplayString,
        ptpbaseClockPortRunningState              PtpClockPortState,
        ptpbaseClockPortRunningRole               PtpClockRoleType,
        ptpbaseClockPortRunningInterfaceIndex     InterfaceIndexOrZero,
        ptpbaseClockPortRunningTransport          AutonomousType,
        ptpbaseClockPortRunningEncapsulationType  AutonomousType,
        ptpbaseClockPortRunningTxMode             PtpClockTxModeType,
        ptpbaseClockPortRunningRxMode             PtpClockTxModeType,
        ptpbaseClockPortRunningPacketsReceived    Counter64,
        ptpbaseClockPortRunningPacketsSent        Counter64
}

ptpbaseClockPortRunningDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
```

```
        logical group of PTP devices."
    ::= { ptpbaseClockPortRunningEntry 1 }

ptpbaseClockPortRunningClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the clock type as defined in the
        Textual convention description."
    ::= { ptpbaseClockPortRunningEntry 2 }

ptpbaseClockPortRunningClockInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockPortRunningEntry 3 }

ptpbaseClockPortRunningPortNumberIndex OBJECT-TYPE
    SYNTAX          PtpClockPortNumber
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP portnumber associated with this
        clock port."
    ::= { ptpbaseClockPortRunningEntry 4 }

ptpbaseClockPortRunningName OBJECT-TYPE
    SYNTAX          DisplayString (SIZE  (1..64))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP clock port name."
    ::= { ptpbaseClockPortRunningEntry 5 }

ptpbaseClockPortRunningState OBJECT-TYPE
    SYNTAX          PtpClockPortState
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the port state returned by PTP engine.

        initializing
        faulty
        disabled
        listening
```

```
        preMaster
        master
        passive
        uncalibrated
        slave        "
    ::= { ptpbaseClockPortRunningEntry 6 }

ptpbaseClockPortRunningRole OBJECT-TYPE
    SYNTAX          PtpClockRoleType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the Clock Role."
    ::= { ptpbaseClockPortRunningEntry 7 }

ptpbaseClockPortRunningInterfaceIndex OBJECT-TYPE
    SYNTAX          InterfaceIndexOrZero
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the interface on the node being used by
        the PTP Clock for PTP communication."
    ::= { ptpbaseClockPortRunningEntry 8 }

ptpbaseClockPortRunningTransport OBJECT-TYPE
    SYNTAX          AutonomousType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the transport protocol being used for PTP
        communication (the mapping used)."
    ::= { ptpbaseClockPortRunningEntry 9 }

ptpbaseClockPortRunningEncapsulationType OBJECT-TYPE
    SYNTAX          AutonomousType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the type of encapsulation if the
        interface is adding extra layers (e.g., VLAN, Pseudowire
        encapsulation...) for the PTP messages."
    ::= { ptpbaseClockPortRunningEntry 10 }

ptpbaseClockPortRunningTxMode OBJECT-TYPE
    SYNTAX          PtpClockTxModeType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the clock transmission mode as
```

```
        unicast:       Using unicast communication channel.
        multicast:     Using Multicast communication channel.
        multicast-mix: Using multicast-unicast communication channel"
    ::= { ptpbaseClockPortRunningEntry 11 }

ptpbaseClockPortRunningRxMode OBJECT-TYPE
    SYNTAX          PtpClockTxModeType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the clock receive mode as

        unicast:       Using unicast communication channel.
        multicast:     Using Multicast communication channel.
        multicast-mix: Using multicast-unicast communication channel"
    ::= { ptpbaseClockPortRunningEntry 12 }

ptpbaseClockPortRunningPacketsReceived OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the packets received on the clock port
        (cumulative). These counters are discontinuous."
    ::= { ptpbaseClockPortRunningEntry 13 }

ptpbaseClockPortRunningPacketsSent OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the packets sent on the clock port
        (cumulative). These counters are discontinuous."
    ::= { ptpbaseClockPortRunningEntry 14 }



ptpbaseClockPortTransDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockPortTransDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the Transparent clock ports running
        dataset for a particular domain."
    ::= { ptpbaseMIBClockInfo 10 }
```

```
ptpbaseClockPortTransDSEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockPortTransDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing clock port Transparent
        dataset information about a single clock port"
    INDEX           {
                        ptpbaseClockPortTransDSDomainIndex,
                        ptpbaseClockPortTransDSInstanceIndex,
                        ptpbaseClockPortTransDSPortNumberIndex
                    }
    ::= { ptpbaseClockPortTransDSTable 1 }

PtpbaseClockPortTransDSEntry ::= SEQUENCE {
        ptpbaseClockPortTransDSDomainIndex        PtpClockDomainType,
        ptpbaseClockPortTransDSInstanceIndex      PtpClockInstanceType,
        ptpbaseClockPortTransDSPortNumberIndex    PtpClockPortNumber,
        ptpbaseClockPortTransDSPortIdentity       PtpClockIdentity,
        ptpbaseClockPortTransDSlogMinPdelayReqInt PtpClockIntervalBase2,
        ptpbaseClockPortTransDSFaultyFlag         TruthValue,
        ptpbaseClockPortTransDSPeerMeanPathDelay  PtpClockTimeInterval
}

ptpbaseClockPortTransDSDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the domain number used to create a
        Logical group of PTP devices."
    ::= { ptpbaseClockPortTransDSEntry 1 }

ptpbaseClockPortTransDSInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockPortTransDSEntry 2 }

ptpbaseClockPortTransDSPortNumberIndex OBJECT-TYPE
    SYNTAX          PtpClockPortNumber
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP port number associated with this
        port."
```

```
    REFERENCE       "Section 7.5.2 Port Identity of [IEEE 1588-2008]"
    ::= { ptpbaseClockPortTransDSEntry 3 }

ptpbaseClockPortTransDSPortIdentity OBJECT-TYPE
    SYNTAX          PtpClockIdentity
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value of the PortIdentity
        attribute of the local port."
    REFERENCE
        "Section 8.3.3.2.1 transparentClockPortDS.portIdentity of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockPortTransDSEntry 4 }

ptpbaseClockPortTransDSlogMinPdelayReqInt OBJECT-TYPE
    SYNTAX          PtpClockIntervalBase2
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value of the logarithm to the
        base 2 of the minPdelayReqInterval."
    REFERENCE
       "Section 8.3.3.3.1 transparentClockPortDS.logMinPdelayReqInterval
        of [IEEE 1588-2008]"
    ::= { ptpbaseClockPortTransDSEntry 5 }

ptpbaseClockPortTransDSFaultyFlag OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the value TRUE if the port is faulty
        and FALSE if the port is operating normally."
    REFERENCE
        "Section 8.3.3.3.2 transparentClockPortDS.faultyFlag of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockPortTransDSEntry 6 }

ptpbaseClockPortTransDSPeerMeanPathDelay OBJECT-TYPE
    SYNTAX          PtpClockTimeInterval
    UNITS           "Time Interval"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies, if the delayMechanism used is P2P, the
        value of the estimate of the current one-way propagation delay,
        i.e., <meanPathDelay> on the link attached to this port,
        computed using the peer delay mechanism. If the value of the
```

```
        delayMechanism used is E2E, then the value will be zero."
    REFERENCE
        "Section 8.3.3.3.3 transparentClockPortDS.peerMeanPathDelay of
        [IEEE 1588-2008]"
    ::= { ptpbaseClockPortTransDSEntry 7 }



ptpbaseClockPortAssociateTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbaseClockPortAssociateEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about a given port's associated ports.

        For a master port: multiple slave ports that have established
                           sessions with the current master port.
        For a slave port:  the list of masters available for a given
                           slave port.

        Session information (packets, errors) to be displayed based on
        availability and scenario."
    ::= { ptpbaseMIBClockInfo 11 }


--
-- Well Known transport types for PTP communication.
--
ptpbaseWellKnownTransportTypes OBJECT IDENTIFIER ::= {
ptpbaseMIBClockInfo 12 }

ptpbaseTransportTypeIPversion4 OBJECT-IDENTITY
    STATUS current
    DESCRIPTION
        "IP version 4"
    ::= { ptpbaseWellKnownTransportTypes 1 }

ptpbaseTransportTypeIPversion6 OBJECT-IDENTITY
   STATUS current
     DESCRIPTION
        "IP version 6"
     ::= { ptpbaseWellKnownTransportTypes 2 }

ptpbaseTransportTypeEthernet OBJECT-IDENTITY
   STATUS current
    DESCRIPTION
        "Ethernet"
     ::= { ptpbaseWellKnownTransportTypes 3 }
```

```
ptpbaseTransportTypeDeviceNET OBJECT-IDENTITY
   STATUS current
    DESCRIPTION
        "Device NET"
     ::= { ptpbaseWellKnownTransportTypes 4 }

ptpbaseTransportTypeControlNET OBJECT-IDENTITY
   STATUS current
    DESCRIPTION
        "Control NET"
     ::= { ptpbaseWellKnownTransportTypes 5 }


ptpbaseTransportTypeIEC61158 OBJECT-IDENTITY
   STATUS current
    DESCRIPTION
        "IEC61158"
     ::= { ptpbaseWellKnownTransportTypes 6 }


--
-- Well Known encapsulation types for PTP communication.
--
ptpbaseWellKnownEncapsulationTypes OBJECT IDENTIFIER ::= {
ptpbaseMIBClockInfo 13 }

ptpbaseEncapsulationTypeEthernet OBJECT-IDENTITY
    STATUS current
    DESCRIPTION
        "Ethernet Encapsulation type."
    ::= { ptpbaseWellKnownEncapsulationTypes 1 }


ptpbaseEncapsulationTypeVLAN OBJECT-IDENTITY
    STATUS current
    DESCRIPTION
        "VLAN Encapsulation type."
    ::= { ptpbaseWellKnownEncapsulationTypes 2 }

ptpbaseEncapsulationTypeUDPIPLSP OBJECT-IDENTITY
   STATUS current
     DESCRIPTION
        "UDP/IP over MPLS Encapsulation type."
     ::= { ptpbaseWellKnownEncapsulationTypes 3 }

ptpbaseEncapsulationTypePWUDPIPLSP OBJECT-IDENTITY
   STATUS current
     DESCRIPTION
        "UDP/IP Pseudowire over MPLS Encapsulation type."
```

```
     ::= { ptpbaseWellKnownEncapsulationTypes 4 }

ptpbaseEncapsulationTypePWEthernetLSP OBJECT-IDENTITY
   STATUS current
     DESCRIPTION
        "Ethernet Pseudowire over MPLS Encapsulation type."
     ::= { ptpbaseWellKnownEncapsulationTypes 5 }



ptpbaseClockPortAssociateEntry OBJECT-TYPE
    SYNTAX          PtpbaseClockPortAssociateEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in the table, containing information about a single
        associated port for the given clockport."
    INDEX           {
                        ptpClockPortCurrentDomainIndex,
                        ptpClockPortCurrentClockTypeIndex,
                        ptpClockPortCurrentClockInstanceIndex,
                        ptpClockPortCurrentPortNumberIndex,
                        ptpbaseClockPortAssociatePortIndex
                    }
    ::= { ptpbaseClockPortAssociateTable 1 }

PtpbaseClockPortAssociateEntry ::= SEQUENCE {
        ptpClockPortCurrentDomainIndex           PtpClockDomainType,
        ptpClockPortCurrentClockTypeIndex        PtpClockType,
        ptpClockPortCurrentClockInstanceIndex    PtpClockInstanceType,
        ptpClockPortCurrentPortNumberIndex       PtpClockPortNumber,
        ptpbaseClockPortAssociatePortIndex       Unsigned32,
        ptpbaseClockPortAssociateAddressType     AutonomousType,
        ptpbaseClockPortAssociateAddress
PtpClockPortTransportTypeAddress,
        ptpbaseClockPortAssociatePacketsSent     Counter64,
        ptpbaseClockPortAssociatePacketsReceived Counter64,
        ptpbaseClockPortAssociateInErrors        Counter64,
        ptpbaseClockPortAssociateOutErrors       Counter64
}

ptpClockPortCurrentDomainIndex OBJECT-TYPE
    SYNTAX          PtpClockDomainType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the given port's domain number."
    ::= { ptpbaseClockPortAssociateEntry 1 }
```

```
ptpClockPortCurrentClockTypeIndex OBJECT-TYPE
    SYNTAX          PtpClockType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the given port's clock type."
    ::= { ptpbaseClockPortAssociateEntry 2 }

ptpClockPortCurrentClockInstanceIndex OBJECT-TYPE
    SYNTAX          PtpClockInstanceType
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the instance of the clock for this clock
        type in the given domain."
    ::= { ptpbaseClockPortAssociateEntry 3 }

ptpClockPortCurrentPortNumberIndex OBJECT-TYPE
    SYNTAX          PtpClockPortNumber
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the PTP Port Number for the given port."
    ::= { ptpbaseClockPortAssociateEntry 4 }

ptpbaseClockPortAssociatePortIndex OBJECT-TYPE
    SYNTAX          Unsigned32 (1..65535)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This object specifies the associated port's serial number in
        the current port's context."
    ::= { ptpbaseClockPortAssociateEntry 5 }

ptpbaseClockPortAssociateAddressType OBJECT-TYPE
    SYNTAX          AutonomousType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the peer port's network address type used
        for PTP communication. The OCTET STRING representation of the
        OID of ptpbaseWellKnownTransportTypes will be used in the values
        contained in the OCTET STRING."
    ::= { ptpbaseClockPortAssociateEntry 6 }

ptpbaseClockPortAssociateAddress OBJECT-TYPE
    SYNTAX          PtpClockPortTransportTypeAddress
    MAX-ACCESS      read-only
    STATUS          current
```

```
    DESCRIPTION
        "This object specifies the peer port's network address used for
        PTP communication."
    ::= { ptpbaseClockPortAssociateEntry 7 }

ptpbaseClockPortAssociatePacketsSent OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of packets sent to this peer port from the current
        port. These counters are discontinuous."
    ::= { ptpbaseClockPortAssociateEntry 8 }

ptpbaseClockPortAssociatePacketsReceived OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of packets received from this peer port by the
        current port. These counters are discontinuous."
    ::= { ptpbaseClockPortAssociateEntry 9 }

ptpbaseClockPortAssociateInErrors OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the input errors associated with the
        peer port. These counters are discontinuous."
    ::= { ptpbaseClockPortAssociateEntry 10 }

ptpbaseClockPortAssociateOutErrors OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           "packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object specifies the output errors associated with the
        peer port. These counters are discontinuous."
    ::= { ptpbaseClockPortAssociateEntry 11 }


-- Conformance Information Definition

ptpbaseMIBCompliances  OBJECT IDENTIFIER
```

```
    ::= { ptpbaseMIBConformance 1 }

ptpbaseMIBGroups  OBJECT IDENTIFIER
    ::= { ptpbaseMIBConformance 2 }


ptpbaseMIBCompliancesSystemInfo MODULE-COMPLIANCE
    STATUS          current
    DESCRIPTION
        "Compliance statement for agents that provide read-only support
        for PTPBASE-MIB to provide system level information of clock
        devices.
        Such devices can only be monitored using this MIB module.

        The Module is implemented with support for read-only.  In other
        words, only monitoring is available by implementing this
        MODULE-COMPLIANCE."
    MODULE          -- this module
    MANDATORY-GROUPS { ptpbaseMIBSystemInfoGroup }
    ::= { ptpbaseMIBCompliances 1 }

ptpbaseMIBCompliancesClockInfo MODULE-COMPLIANCE
    STATUS          current
    DESCRIPTION
        "Compliance statement for agents that provide read-only support
        for PTPBASE-MIB to provide clock related information.
        Such devices can only be monitored using this MIB module.

        The Module is implemented with support for read-only. In other
        words, only monitoring is available by implementing this
        MODULE-COMPLIANCE."
    MODULE          -- this module
    MANDATORY-GROUPS {
                        ptpbaseMIBClockCurrentDSGroup,
                        ptpbaseMIBClockParentDSGroup,
                        ptpbaseMIBClockDefaultDSGroup,
                        ptpbaseMIBClockRunningGroup,
                        ptpbaseMIBClockTimepropertiesGroup
                     }
    ::= { ptpbaseMIBCompliances 2 }

ptpbaseMIBCompliancesClockPortInfo MODULE-COMPLIANCE
    STATUS          current
    DESCRIPTION
        "Compliance statement for agents that provide read-only support
        for PTPBASE-MIB to provide clock port related information.
        Such devices can only be monitored using this MIB module.

        The Module is implemented with support for read-only. In other
```

```
        words, only monitoring is available by implementing this
        MODULE-COMPLIANCE."
    MODULE          -- this module
    MANDATORY-GROUPS {
                        ptpbaseMIBClockPortGroup,
                        ptpbaseMIBClockPortDSGroup,
                        ptpbaseMIBClockPortRunningGroup,
                        ptpbaseMIBClockPortAssociateGroup
                    }
    ::= { ptpbaseMIBCompliances 3 }

ptpbaseMIBCompliancesTransparentClockInfo MODULE-COMPLIANCE
    STATUS          current
    DESCRIPTION
        "Compliance statement for agents that provide read-only support
        for PTPBASE-MIB to provide Transparent clock related
        information.
        Such devices can only be monitored using this MIB module.

        The Module is implemented with support for read-only. In other
        words, only monitoring is available by implementing this
        MODULE-COMPLIANCE."
    MODULE          -- this module
    MANDATORY-GROUPS {
                        ptpbaseMIBClockTranparentDSGroup,
                        ptpbaseMIBClockPortTransDSGroup
                    }
    ::= { ptpbaseMIBCompliances 4 }

ptpbaseMIBSystemInfoGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseSystemDomainTotals,
                        ptpDomainClockPortsTotal,
                        ptpbaseSystemProfile
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing system-wide
        information"
    ::= { ptpbaseMIBGroups 1 }

ptpbaseMIBClockCurrentDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockCurrentDSStepsRemoved,
                        ptpbaseClockCurrentDSOffsetFromMaster,
                        ptpbaseClockCurrentDSMeanPathDelay
                    }
    STATUS          current
    DESCRIPTION
```

```
        "Group which aggregates objects describing PTP Current Dataset
        information"
    ::= { ptpbaseMIBGroups 2 }

ptpbaseMIBClockParentDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockParentDSParentPortIdentity,
                        ptpbaseClockParentDSParentStats,
                        ptpbaseClockParentDSOffset,
                        ptpbaseClockParentDSClockPhChRate,
                        ptpbaseClockParentDSGMClockIdentity,
                        ptpbaseClockParentDSGMClockPriority1,
                        ptpbaseClockParentDSGMClockPriority2,
                        ptpbaseClockParentDSGMClockQualityClass,
                        ptpbaseClockParentDSGMClockQualityAccuracy,
                        ptpbaseClockParentDSGMClockQualityOffset
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP Parent Dataset
        information"
    ::= { ptpbaseMIBGroups 3 }

ptpbaseMIBClockDefaultDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockDefaultDSTwoStepFlag,
                        ptpbaseClockDefaultDSClockIdentity,
                        ptpbaseClockDefaultDSPriority1,
                        ptpbaseClockDefaultDSPriority2,
                        ptpbaseClockDefaultDSSlaveOnly,
                        ptpbaseClockDefaultDSQualityClass,
                        ptpbaseClockDefaultDSQualityAccuracy,
                        ptpbaseClockDefaultDSQualityOffset
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP Default Dataset
        information"
    ::= { ptpbaseMIBGroups 4 }

ptpbaseMIBClockRunningGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockRunningState,
                        ptpbaseClockRunningPacketsSent,
                        ptpbaseClockRunningPacketsReceived
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP running state
```

```
        information"
    ::= { ptpbaseMIBGroups 5 }

ptpbaseMIBClockTimepropertiesGroup OBJECT-GROUP
    OBJECTS  {
                ptpbaseClockTimePropertiesDSCurrentUTCOffsetValid,
                ptpbaseClockTimePropertiesDSCurrentUTCOffset,
                ptpbaseClockTimePropertiesDSLeap59,
                ptpbaseClockTimePropertiesDSLeap61,
                ptpbaseClockTimePropertiesDSTimeTraceable,
                ptpbaseClockTimePropertiesDSFreqTraceable,
                ptpbaseClockTimePropertiesDSPTPTimescale,
                ptpbaseClockTimePropertiesDSSource
             }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP Time Properties
        information"
    ::= { ptpbaseMIBGroups 6 }

ptpbaseMIBClockTranparentDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockTransDefaultDSClockIdentity,
                        ptpbaseClockTransDefaultDSNumOfPorts,
                        ptpbaseClockTransDefaultDSDelay,
                        ptpbaseClockTransDefaultDSPrimaryDomain
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP Transparent
        Dataset
        information"
    ::= { ptpbaseMIBGroups 7 }

ptpbaseMIBClockPortGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockPortName,
                        ptpbaseClockPortSyncTwoStep,
                        ptpbaseClockPortCurrentPeerAddress,
                        ptpbaseClockPortNumOfAssociatedPorts,
                        ptpbaseClockPortCurrentPeerAddressType,
                        ptpbaseClockPortRole
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing information for a
        given PTP Port."
    ::= { ptpbaseMIBGroups 8 }
```

```
ptpbaseMIBClockPortDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockPortDSName,
                        ptpbaseClockPortDSPortIdentity,
                        ptpbaseClockPortDSlogAnnouncementInterval,
                        ptpbaseClockPortDSAnnounceRctTimeout,
                        ptpbaseClockPortDSlogSyncInterval,
                        ptpbaseClockPortDSMinDelayReqInterval,
                        ptpbaseClockPortDSPeerDelayReqInterval,
                        ptpbaseClockPortDSDelayMech,
                        ptpbaseClockPortDSPeerMeanPathDelay,
                        ptpbaseClockPortDSGrantDuration,
                        ptpbaseClockPortDSPTPVersion
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP Port Dataset
        information"
    ::= { ptpbaseMIBGroups 9 }

ptpbaseMIBClockPortRunningGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockPortRunningName,
                        ptpbaseClockPortRunningState,
                        ptpbaseClockPortRunningRole,
                        ptpbaseClockPortRunningInterfaceIndex,
                        ptpbaseClockPortRunningTransport,
                        ptpbaseClockPortRunningEncapsulationType,
                        ptpbaseClockPortRunningTxMode,
                        ptpbaseClockPortRunningRxMode,
                        ptpbaseClockPortRunningPacketsReceived,
                        ptpbaseClockPortRunningPacketsSent
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP running interface
        information"
    ::= { ptpbaseMIBGroups 10 }

ptpbaseMIBClockPortTransDSGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockPortTransDSPortIdentity,
                        ptpbaseClockPortTransDSlogMinPdelayReqInt,
                        ptpbaseClockPortTransDSFaultyFlag,
                        ptpbaseClockPortTransDSPeerMeanPathDelay
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing PTP TransparentDS
```

```
        information"
    ::= { ptpbaseMIBGroups 11 }

ptpbaseMIBClockPortAssociateGroup OBJECT-GROUP
    OBJECTS         {
                        ptpbaseClockPortAssociatePacketsSent,
                        ptpbaseClockPortAssociatePacketsReceived,
                        ptpbaseClockPortAssociateAddress,
                        ptpbaseClockPortAssociateAddressType,
                        ptpbaseClockPortAssociateInErrors,
                        ptpbaseClockPortAssociateOutErrors
                    }
    STATUS          current
    DESCRIPTION
        "Group which aggregates objects describing information on peer
        PTP ports for a given PTP clock-port."
    ::= { ptpbaseMIBGroups 12 }


END
```

5. Security Considerations

There are no management objects defined in this MIB module that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB module is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB module via direct SNMP SET operations.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP.

The following objects all have a MAX-ACCESS of read-only:

ptpDomainClockPortsTotal,
ptpbaseSystemDomainTotals,
ptpbaseSystemProfile expose general information about the clock system.

ptpbaseClockRunningState,
ptpbaseClockRunningPacketsSent,
ptpbaseClockRunningPacketsReceived expose a clock’s current running status.

ptpbaseClockCurrentDSStepsRemoved,

ptpbaseClockCurrentDSOffsetFromMaster,
ptpbaseClockCurrentDSMeanPathDelay expose the values of a clock's current dataset (currentDS).

ptpbaseClockParentDSParentPortIdentity,
ptpbaseClockParentDSParentStats,
ptpbaseClockParentDSOffset,
ptpbaseClockParentDSClockPhChRate,
ptpbaseClockParentDSGMClockIdentity,
ptpbaseClockParentDSGMClockPriority1,
ptpbaseClockParentDSGMClockPriority2,
ptpbaseClockParentDSGMClockQualityClass,
ptpbaseClockParentDSGMClockQualityAccuracy,
ptpbaseClockParentDSGMClockQualityOffset expose the values of a clock's parent dataset (parentDS).

ptpbaseClockDefaultDSTwoStepFlag,
ptpbaseClockDefaultDSClockIdentity,
ptpbaseClockDefaultDSPriority1,
ptpbaseClockDefaultDSPriority2,
ptpbaseClockDefaultDSSlaveOnly,
ptpbaseClockDefaultDSQualityClass,
ptpbaseClockDefaultDSQualityAccuracy,
ptpbaseClockDefaultDSQualityOffset expose the values of a clock's default dataset (defaultDS).

ptpbaseClockTimePropertiesDSCurrentUTCOffsetValid,
ptpbaseClockTimePropertiesDSCurrentUTCOffset,
ptpbaseClockTimePropertiesDSLeap59,
ptpbaseClockTimePropertiesDSLeap61,
ptpbaseClockTimePropertiesDSTimeTraceable,
ptpbaseClockTimePropertiesDSFreqTraceable,
ptpbaseClockTimePropertiesDSPTPTimescale,
ptpbaseClockTimePropertiesDSSource expose the values of a clock's time properties dataset (timePropertiesDS).

ptpbaseClockTransDefaultDSClockIdentity,
ptpbaseClockTransDefaultDSNumOfPorts,
ptpbaseClockTransDefaultDSDelay,
ptpbaseClockTransDefaultDSPrimaryDomain expose the values of a transparent clock's default dataset (transparentClockDefaultDS).

ptpbaseClockPortName,
ptpbaseClockPortRole,
ptpbaseClockPortSyncTwoStep,
ptpbaseClockPortCurrentPeerAddressType,
ptpbaseClockPortCurrentPeerAddress,
ptpbaseClockPortNumOfAssociatedPorts expose general information about a clock port.

ptpbaseClockPortRunningName,
ptpbaseClockPortRunningState,
ptpbaseClockPortRunningRole,
ptpbaseClockPortRunningInterfaceIndex,
ptpbaseClockPortRunningTransport,
ptpbaseClockPortRunningEncapsulationType,
ptpbaseClockPortRunningTxMode,
ptpbaseClockPortRunningRxMode,
ptpbaseClockPortRunningPacketsReceived,
ptpbaseClockPortRunningPacketsSent expose a clock port's current running status.

ptpbaseClockPortDSName,
ptpbaseClockPortDSPortIdentity,
ptpbaseClockPortDSlogAnnouncementInterval,
ptpbaseClockPortDSAnnounceRctTimeout,
ptpbaseClockPortDSlogSyncInterval,
ptpbaseClockPortDSMinDelayReqInterval,
ptpbaseClockPortDSPeerDelayReqInterval,
ptpbaseClockPortDSDelayMech,
ptpbaseClockPortDSPeerMeanPathDelay,
ptpbaseClockPortDSGrantDuration,
ptpbaseClockPortDSPTPVersion expose the values of a clock port's port dataset (portDS).

ptpbaseClockPortTransDSPortIdentity,
ptpbaseClockPortTransDSlogMinPdelayReqInt,
ptpbaseClockPortTransDSFaultyFlag,
ptpbaseClockPortTransDSPeerMeanPathDelay expose the values of a transparent clock port's port dataset (transparentClockPortDS).

ptpbaseClockPortAssociateAddressType,
ptpbaseClockPortAssociateAddress,
ptpbaseClockPortAssociatePacketsSent,
ptpbaseClockPortAssociatePacketsReceived,
ptpbaseClockPortAssociateInErrors,
ptpbaseClockPortAssociateOutErrors expose information about a clock port's peer node.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET (read) the objects in this MIB module.

Implementations SHOULD provide the security features described by the SNMPv3 framework (see [RFC 3410]), and implementations claiming compliance to the SNMPv3 standard MUST include full support for authentication and privacy via the User-based Security Model (USM) [RFC 3414] with the AES cipher algorithm [RFC 3826]. Implementations

MAY also provide support for the Transport Security Model (TSM) [RFC 5591] in combination with a secure transport such as SSH [RFC 5592] or TLS/DTLS [RFC 6353].

Further, deployment of SNMP versions prior to SNMPv3 is NOT recommended. Instead, it is recommended to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to those objects only to those principals (users) that have legitimate rights to access them.

6. IANA Considerations

The MIB module defined in this document uses the following IANA-assigned OBJECT IDENTIFIER value recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
ptpbaseMIB	{ mib-2 xxx }

[NOTE for IANA: Please allocate an object identifier at http://www.iana.org/assignments/smi-numbers for object ptpbaseMIB.]

7. References

7.1. Normative References

[IEEE 1588-2008] "IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std. 1588(TM)-2008, 24 July 2008

7.2. Informative References

[RFC 1155] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990

[RFC 1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[RFC 1212] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991

[RFC 1215] M. Rose, "A Convention for Defining Traps for use with the

SNMP", RFC 1215, Performance Systems International, March 1991

[RFC 1901] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC 1906] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC 2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119 Harvard University, March 1997.

[RFC 2578] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC 2579] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC 2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC 3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet Standard Management Framework", RFC 3410 SNMP Research, Inc., Network Associates Laboratories, Ericsson, December 2002.

[RFC 3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, Enterasys Networks, BMC Software, Inc., Lucent Technologies, December 2002

[RFC 3412] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3412, SNMP Research, Inc., Enterasys Networks, BMC Software, Inc., Lucent Technologies, December 2002.

[RFC 3413] Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, Nortel Networks, Secure Computing Corporation, December 2002.

[RFC 3414] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, Lucent Technologies, December 2002.

[RFC 3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, Lucent Technologies, BMC Software, Inc., Cisco Systems, Inc., December 2002.

[RFC 3416] Presuhn, R. (Ed.), "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, BMC Software, Inc., December 2002.

[RFC 3417] Presuhn, R. (Ed.), "Transport Mappings for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3417, BMC Software, Inc., December 2002.

[RFC 3826] Blumenthal, U., Maino, F, and K. McCloghrie, "The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model", RFC 3826, Lucent Technologies, Andiamo Systems, Inc., Cisco Systems, Inc., June 2004.

[RFC 5591] Harrington, D., and W. Hardraker, "Transport Security Model for the Simple Network Management Protocol (SNMP)", RFC 5591, Huawei Technologies (USA), Cobham Analytic Solutions, June 2009.

[RFC 5592] Harrington, D., Salowey, J., and W. Hardraker, "Secure Shell Transport Model for the Simple Network Management Protocol (SNMP) ", RFC 5592, Huawei Technologies (USA), Cisco Systems, Cobham Analytic Solutions, June 2009.

[RFC 5905] David L. Mills, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, University of Delaware, June 2010.

[RFC 6353] Hardraker, W., "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", RFC 6353, SPARTA, Inc., July 2011.

[IEEE 802.3-2012] "IEEE Standard for Ethernet", IEEE Std. 802.3 - 2015, 3 September 2015

[G.8265.1] "Precision time protocol telecom profile for frequency synchronization", ITU-T Recommendation G.8265.1, July 2014.

8. Acknowledgements

Thanks to John Linton and Danny Lee for valuable comments, and to Bert Wijnen, Kevin Gross, Alan Luchuk, Chris Elliot, Brian Haberman and Dan Romascanu for their reviews of this MIB module.

9. Author’s Addresses

Vinay Shankarkumar
Cisco Systems,
7100-9 Kit Creek Road,
Research Triangle Park,
NC 27709,
USA.

Email: vinays@cisco.com

Laurent Montini,
Cisco Systems,
11, rue Camille Desmoulins,
92782 Issy-les-Moulineaux,
France.

Email: lmontini@cisco.com

Tim Frost,
Calnex Solutions Ltd.,
Oracle Campus,
Linlithgow,
EH49 7LR,
UK.

Email: tim.frost@calnexsol.com

Greg Dowd,
Microsemi Inc.,
3870 North First Street,
San Jose,
CA 95134,
USA.

Email: greg.dowd@microsemi.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 13, 2017

N. Wu
Huawei
A. Kumar S N
RtBrick Inc.
Y. Zhao
Ericsson
D. Dhody
A. Sinha
Huawei

May 12, 2017

A YANG Data Model for NTP
draft-wu-ntp-ntp-cfg-03

Abstract

This document defines a YANG data model for Network Time Protocol implementations. The data model includes configuration data and state data.

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].

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 http://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 November 13, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Operational State 3
1.2. Terminology 3
1.3. Tree Diagrams 3
2. NTP data model 3
3. Relationship with NTPv4-MIB 6
4. Relationship with RFC7317 7
5. NTP YANG Module 8
6. IANA Considerations 25
7. Security Considerations 26
8. Acknowledgments 26
9. References 26
9.1. Normative References 26
9.2. Informative References 27
Authors' Addresses 28

1. Introduction

This document defines a YANG [RFC6020] data model for Network Time Protocol [RFC5905] implementations.

The data model convers configuration of system parameters of NTP, such as access rules, authentication and VRF binding, and also associations of NTP in different modes and parameters of per-interface. It also provides information about running state of NTP implementations.

1.1. Operational State

NTP Operational State is included in the same tree as NTP configuration, consistent with Network Management Datastore Architecture [I-D.ietf-netmod-revised-datastores]. NTP current state and statistics are also maintained in the operational state. Additionally, the operational state also include the associations state.

1.2. Terminology

1.3. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. NTP data model

This document defines the YANG module "ietf-ntp", which has the following structure:

```
module: ietf-ntp
    +--rw ntp!
    |  +--rw port?              uint16
    |  +--rw refclock-master!
    |  |  +--rw master-stratum?   ntp-stratum
    |  +--rw authentication
    |  |  +--rw auth-enabled?          boolean
    |  |  +--rw trusted-keys* [key-id]
    |  |  |  +--rw key-id    leafref
    |  |  +--rw authentication-keys* [key-id]
    |  |     +--rw key-id       uint32
    |  |     +--rw algorithm?   identityref
```

```
|  |     +--rw password?    ianach:crypt-hash
|  +--rw access-rules
|  |  +--rw access-rule* [access-mode]
|  |     +--rw access-mode    access-modes
|  |     +--rw acl?           -> /acl:access-lists/acl/acl-name
|  +--ro clock-state
|  |  +--ro system-status
|  |     +--ro clock-state?       ntp-clock-status
|  |     +--ro clock-stratum?     ntp-stratum
|  |     +--ro clock-refid?       union
|  |     +--ro nominal-freq?      decimal64
|  |     +--ro actual-freq?       decimal64
|  |     +--ro clock-precision?   uint8
|  |     +--ro clock-offset?      decimal64
|  |     +--ro root-delay?        decimal64
|  |     +--ro root-dispersion?   decimal64
|  |     +--ro peer-dispersion?   decimal64
|  |     +--ro reference-time?    yang:date-and-time
|  |     +--ro sync-state?        ntp-sync-state
|  +--rw associations* [address association-type]
|  |  +--rw address             inet:host
|  |  +--rw association-type    association-modes
|  |  +--rw authentication
|  |  |  +--rw (authentication-type)?
|  |  |     +--:(symmetric-key)
|  |  |        +--rw key-id?   leafref
|  |  +--rw prefer?             boolean
|  |  +--rw burst?              boolean
|  |  +--rw iburst?             boolean
|  |  +--rw source?             if:interface-ref
|  |  +--rw minpoll?            ntp-minpoll
|  |  +--rw maxpoll?            ntp-maxpoll
|  |  +--rw port?               uint16
|  |  +--rw version?            ntp-version
|  |  +--ro stratum?            ntp-stratum
|  |  +--ro refid?              union
|  |  +--ro reach?              uint8
|  |  +--ro unreach?            uint8
|  |  +--ro poll?               uint8
|  |  +--ro now?                uint32
|  |  +--ro offset?             decimal64
|  |  +--ro delay?              decimal64
|  |  +--ro dispersion?         decimal64
|  |  +--ro originate-time?     yang:date-and-time
|  |  +--ro receive-time?       yang:date-and-time
|  |  +--ro transmit-time?      yang:date-and-time
|  |  +--ro input-time?         yang:date-and-time
|  |  +--ro ntp-statistics
```

```
|  |     +--ro packet-sent?        yang:counter32
|  |     +--ro packet-sent-fail?   yang:counter32
|  |     +--ro packet-received?    yang:counter32
|  |     +--ro packet-dropped?     yang:counter32
|  +--rw interface
|     +--rw interface* [interface]
|        +--rw interface           if:interface-ref
|        +--rw broadcast-server!
|        |  +--rw ttl?              uint8
|        |  +--rw authentication
|        |  |  +--rw (authentication-type)?
|        |  |     +--:(symmetric-key)
|        |  |        +--rw key-id?   leafref
|        |  +--rw minpoll?          ntp-minpoll
|        |  +--rw maxpoll?          ntp-maxpoll
|        |  +--rw port?             uint16
|        |  +--rw version?          ntp-version
|        +--rw broadcast-client!
|        +--rw multicast-server* [address]
|        |  +--rw address
|        |  |       rt-types:ip-multicast-group-address
|        |  +--rw ttl?              uint8
|        |  +--rw authentication
|        |  |  +--rw (authentication-type)?
|        |  |     +--:(symmetric-key)
|        |  |        +--rw key-id?   leafref
|        |  +--rw minpoll?          ntp-minpoll
|        |  +--rw maxpoll?          ntp-maxpoll
|        |  +--rw port?             uint16
|        |  +--rw version?          ntp-version
|        +--rw multicast-client* [address]
|        |  +--rw address    rt-types:ip-multicast-group-address
|        +--rw manycast-server* [address]
|        |  +--rw address    rt-types:ip-multicast-group-address
|        +--rw manycast-client* [address]
|           +--rw address
|           |       rt-types:ip-multicast-group-address
|           +--rw authentication
|           |  +--rw (authentication-type)?
|           |     +--:(symmetric-key)
|           |        +--rw key-id?   leafref
|           +--rw ttl?              uint8
|           +--rw minclock?         uint8
|           +--rw maxclock?         uint8
|           +--rw beacon?           uint8
|           +--rw minpoll?          ntp-minpoll
|           +--rw maxpoll?          ntp-maxpoll
|           +--rw port?             uint16
```

```
|           +--rw version?          ntp-version
+--ro ntp-statistics
   +--ro packet-sent?        yang:counter32
   +--ro packet-sent-fail?   yang:counter32
   +--ro packet-received?    yang:counter32
   +--ro packet-dropped?     yang:counter32
```

This data model defines two primary containers, one for NTP configuration and the other is for NTP running state. The NTP configuration container includes data nodes for access rules, authentication, associations and interfaces. In the NTP running state container, there are data nodes for system status and associations.

3. Relationship with NTPv4-MIB

If the device implements the NTPv4-MIB [RFC5907], data nodes in container ntp and ntp-state from YANG module can be mapped to table entries in NTPv4-MIB.

The following tables list the YANG data nodes with corresponding objects in the NTPv4-MIB.

YANG data nodes in /ntp/	NTPv4-MIB objects
ntp-enabled	ntpEntStatusCurrentMode

YANG data nodes in /ntp/associations	NTPv4-MIB objects
address	ntpAssocAddressType
	ntpAssocAddress

YANG NTP Configuration Data Nodes and Related NTPv4-MIB Objects

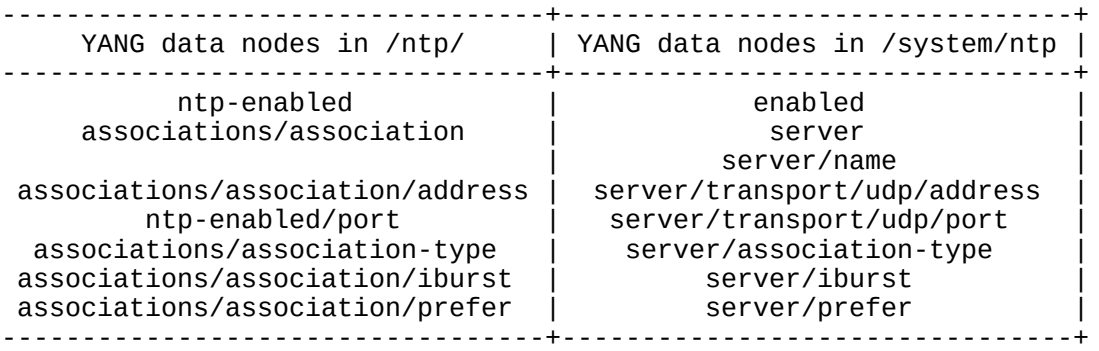
YANG data nodes in /ntp-state /system-status	NTPv4-MIB objects
clock-state	ntpEntStatusCurrentMode
clock-stratum	ntpEntStatusStratum
clock-refid	ntpEntStatusActiveRefSourceId
	ntpEntStatusActiveRefSourceName
clock-precision	ntpEntTimePrecision
clock-offset	ntpEntStatusActiveOffset
root-dispersion	ntpEntStatusDispersion

YANG data nodes in /ntp-state /associations-status/association-status/	NTPv4-MIB objects
association-source	ntpAssocAddressType
	ntpAssocAddress
association-stratum	ntpAssocStratum
association-refid	ntpAssocRefId
association-offset	ntpAssocOffset
association-delay	ntpAssocStatusDelay
association-dispersion	ntpAssocStatusDispersion
association-sent	ntpAssocStatOutPkts
association-received	ntpAssocStatInPkts
association-dropped	ntpAssocStatProtocolError

YANG NTP State Data Nodes and Related NTPv4-MIB Objects

4. Relationship with RFC7317

This section describes the relationship with NTP definition in Section 3.2 System Time Management of [RFC7317] . YANG data nodes in /ntp/ also supports interface related configurations which is not supported in /system/ntp



YANG NTP Configuration Data Nodes and counterparts in RFC7317 Objects

5. NTP YANG Module

```
<CODE BEGINS> file "ietf-ntp@2017-05-12.yang"
module ietf-ntp {

  namespace "urn:ietf:params:xml:ns:yang:ietf-ntp";

  prefix "ntp";

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-interfaces {
    prefix "if";
  }

  import iana-crypt-hash {
    prefix ianach;
  }

  import ietf-key-chain {
    prefix "key-chain";
  }

  import ietf-access-control-list {
    prefix "acl";
  }
```

```
import ietf-routing-types {
  prefix "rt-types";
}

organization
  "IETF NTP (Network Time Protocol) Working Group";

contact
  "WG Web:  <http://tools.ietf.org/wg/ntp/>
   WG List:  <mailto: ntpwg@lists.ntp.org
   WG Chair: Karen O'Donoghue
             <mailto: odonoghue@isoc.org>
   Editor:   Eric Wu
             <mailto:eric.wu@huawei.com>
   Editor:   Anil Kumar S N
             <mailto:anil.ietf@gmail.com>
   Editor:   Yi Zhao
             <mailto:yi.z.zhao@ericsson.com>
   Editor:   Dhruv Dhody
             <mailto:dhruv.ietf@gmail.com>
   Editor:   Ankit Kumar Sinha
             <mailto:ankit.ietf@gmail.com>";
description
  "This YANG module defines essential components for the
   management of a routing subsystem.

   Copyright (c) 2017 IETF Trust and the persons identified
   as authors of the code. All rights reserved.

   Redistribution and use in source and binary forms,
   with or without modification, is permitted pursuant to,
   and subject to the license terms contained in, the
   Simplified BSD License set forth in Section 4.c of the
   IETF Trust's Legal Provisions Relating to IETF Documents
   (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX;
   see the RFC itself for full legal notices.";

revision 2017-05-12 {
  description
    "Updated revision.";
  reference
    "RFC XXXX: A YANG Data Model for NTP Management";
}

/* Typedef Definitions */
```

```
      typedef ntp-stratum {
        type uint8 {
          range "1..16";
        }
        description
          "The level of each server in the hierarchy is defined by
          a stratum number. Primary servers are assigned stratum
          one; secondary servers at each lower level are assigned
          stratum numbers one greater than the preceding level";
      }

      typedef ntp-version {
        type uint8 {
          range "1..4";
        }
        default "3";
        description
          "The current NTP version supported by corresponding
          association.";
      }

      typedef ntp-minpoll {
        type uint8 {
          range "4..17";
        }
        default "6";
        description
          "The minimum poll exponent for this NTP association.";
      }


      typedef ntp-maxpoll {
        type uint8 {
          range "4..17";
        }
        default "10";
        description
          "The maximul poll exponent for this NTP association.";
      }

      typedef access-modes {
        type enumeration {
          enum peer {
            value "0";
            description
              "Sets the fully access authority. Both time
              request and control query can be performed
              on the local NTP service, and the local clock
```

```
            can be synchronized to the remote server.";
        }
        enum server {
          value "1";
          description
            "Enables the server access and query.
            Both time requests and control query can be
            performed on the local NTP service, but the
            local clock cannot be synchronized to the
            remote server.";
        }
        enum synchronization {
          value "2";
          description
            "Enables the server to access.
            Only time request can be performed on the
            local NTP service.";
        }
        enum query {
          value "3";
          description
            "Sets the maximum access limitation.
            Control query can be performed only on the
            local NTP service.";
        }
      }
      description
        "This defines NTP acess modes.";
    }

    typedef association-modes {
      type enumeration {
        enum server {
          value "0";
          description
            "Use client association mode.  This device
            will not provide synchronization to the
            configured NTP server.";
        }
        enum peer {
          value "1";
          description
            "Use symmetric active association mode.
            This device may provide synchronization
            to the configured NTP server.";
        }
        enum pool {
          value "2";
```

```
            description
              "Use client association mode with one or
              more of the NTP servers found by DNS
              resolution of the domain name given by
              the 'address' leaf.  This device will not
              provide synchronization to the servers.";
          }
        }
        description
          "This defines NTP association modes.";
      }

      typedef ntp-clock-status {
        type enumeration {
          enum synchronized {
            value "0";
            description
              "Indicates that the local clock has been
              synchronized with an NTP server or
              the reference clock.";
          }
          enum unsynchronized {
            value "1";
            description
              "Indicates that the local clock has not been
              synchronized with any NTP server.";
          }
        }
        description
          "This defines NTP clock status.";
      }

      typedef ntp-sync-state {
        type enumeration {
          enum clock-not-set {
            value "0";
            description
              "Indicates the clock is not updated.";
          }
          enum freq-set-by-cfg {
            value "1";
            description
              "Indicates the clock frequency is set by
              NTP configuration.";
          }
          enum clock-set {
            value "2";
            description
```

```
            "Indicates the clock is set.";
        }
        enum freq-not-determined {
          value "3";
          description
            "Indicates the clock is set but the frequency
            is not determined.";
        }
        enum clock-synchronized {
          value "4";
          description
            "Indicates that the clock is synchronized";
        }
        enum spike {
          value "5";
          description
            "Indicates a time difference of more than 128
            milliseconds is detected between NTP server
            and client clock. The clock change will take
            effect in XXX seconds.";
        }
      }
      description
        "This defines NTP clock sync states.";
    }

    /* Groupings */
    grouping authentication-key {
      description
        "To define an authentication key for a Network Time
        Protocol (NTP) time source.";
      leaf key-id {
        type uint32 {
          range "1..max";
        }
        description
          "Authentication key identifier.";
      }
      leaf algorithm {
        type identityref {
          base key-chain:crypto-algorithm;
        }
        description
          "Authentication algorithm.";
      }
      leaf password {
        type ianach:crypt-hash;
        description "Clear or encrypted mode for password text.";
```

```
    }
  }

  grouping authentication-type-param {
    description
        "Authentication type.";
    choice authentication-type {
      description
        "Type of authentication.";
      case symmetric-key {
        leaf key-id {
          type leafref {
            path "/ntp:ntp/ntp:authentication/"
            + "ntp:authentication-keys/ntp:key-id";
          }
          description
            "Authentication key id referenced in this
            association.";
        }
      }
    }
  }

  grouping statistics {
    description
      "NTP packet statistic.";
    leaf packet-sent {
      type yang:counter32;
      description
        "Indicates the total number of packets sent.";
    }
    leaf packet-sent-fail {
      type yang:counter32;
      description
        "Indicates the number of times packet
        sending failed.";
    }
    leaf packet-received {
      type yang:counter32;
      description
        "Indicates the total number of packets received.";
    }
    leaf packet-dropped {
      type yang:counter32;
      description
        "Indicates the number of packets dropped.";
    }
  }
```

```
      grouping comman-attributes {
        description
          "NTP common attributes for configuration.";
        leaf minpoll {
          type ntp-minpoll;
          description
            "The minimum poll interval used in this association.";
        }
        leaf maxpoll {
          type ntp-maxpoll;
          description
            "The maximul poll interval used in this association.";
        }
        leaf port {
          type uint16 {
            range "123 | 1025..max";
          }
          default "123";
          description
            "Specify the port used to send NTP packets.";
        }
        leaf version {
          type ntp-version;
          description
            "NTP version.";
        }
      }

      /* Configuration data nodes */
      container ntp {
        presence
          "NTP is enable";
        description
          "Configuration parameters for NTP.";
        leaf port {
          type uint16 {
            range "123 | 1025..max";
          }
          default "123";
          description
            "Specify the port used to send NTP packets.";
        }

     container refclock-master {
       presence
          "NTP master clock is enable";
       description
         "Configures the device as NTP server.";
```

```
          leaf master-stratum {
            type ntp-stratum;
            default "16";
            description
              "Stratum level from which NTP
              clients get their time synchronized.";
      }
    }
    container authentication {
      description
         "Configuration of authentication.";
         leaf auth-enabled {
           type boolean;
           default false;
           description
             "Controls whether NTP authentication is enabled
             or disabled on this device.";
         }
         list trusted-keys {
           key "key-id";
           description
             "List of keys trusted by NTP.";
           leaf key-id {
             type leafref {
               path "/ntp:ntp/ntp:authentication/"
                   + "ntp:authentication-keys/ntp:key-id";
             }
             description
             "The key trusted by NTP.";
           }
         }
         list authentication-keys {
           key "key-id";
           uses authentication-key;
           description
             "List of authentication key.";
         }
    }

    container access-rules {
      description
        "Configuration of access rules.";
         list access-rule {
           key "access-mode";
           description
             "List of access rules.";
           leaf access-mode {
             type access-modes;
```

```
              description
                "NTP access mode.";
              }
             leaf acl {
               type leafref {
                path "/acl:access-lists/acl:acl/acl:acl-name";
               }
               description
                 "NTP ACL.";
              }
            }
      }

      container clock-state {
        config "false";
        description
          "Operational state of the NTP.";

        container system-status {
          description
             "System status of NTP.";
          leaf clock-state {
            type ntp-clock-status;
            description "Indicates the state of system clock.";
          }
          leaf clock-stratum {
              type ntp-stratum;
              description
                "Indicates the stratum of the reference clock.";
          }
          leaf clock-refid {
              type union {
                  type inet:ipv4-address;
                  type binary {
                      length "4";
                  }
                  type string {
                      length "4";
                  }
              }
              description
                "IPv4 address or first 32 bits of the MD5 hash of
                the IPv6 address or reference clock of the peer to
                which clock is synchronized.";
          }
          leaf nominal-freq {
              type decimal64 {
                  fraction-digits 4;
```

```
            }
            description
              "Indicates the nominal frequency of the
              local clock, in Hz.";
          }
          leaf actual-freq {
            type decimal64 {
                fraction-digits 4;
            }
            description
              "Indicates the actual frequency of the
              local clock, in Hz.";
          }
          leaf clock-precision {
            type uint8;
            description
              "Precision of the clock of this system
              in Hz.(prec=2^(-n))";
          }
          leaf clock-offset {
            type decimal64 {
                fraction-digits 4;
            }
            description
              "Offset of clock to synchronized peer,
              in milliseconds.";
          }
          leaf root-delay {
            type decimal64 {
                fraction-digits 2;
            }
            description
              "Total delay along path to root clock,
              in milliseconds.";
          }
          leaf root-dispersion {
            type decimal64 {
                fraction-digits 2;
            }
            description
              "Indicates the dispersion between the local clock
              and the master reference clock, in milliseconds.";
          }
          leaf peer-dispersion {
            type decimal64 {
                fraction-digits 2;
            }
            description
```

```
              "Indicates the dispersion between the local clock
              and the peer clock, in milliseconds.";
          }
          leaf reference-time {
              type yang:date-and-time;
              description
                "Indicates reference timestamp.";
          }
          leaf sync-state {
            type ntp-sync-state;
            description
              "Indicates the synchronization status of
              the local clock.";
          }
        }
      }

      list associations {
        key "address association-type";
        description
          "list of association.";
        leaf address {
          type inet:host;
          description
            "The address of this association.";
        }
        leaf association-type {
          type association-modes;
          description
            "The desired association type for this NTP server.";
        }
        container authentication{
          description
            "Authentication type.";
          uses authentication-type-param;
        }
        leaf prefer {
          type boolean;
          default "false";
          description
            "Whether this association is preferred.";
        }
        leaf burst {
          type boolean;
          default "false";
          description
            "Sends a series of packets instead of a single packet
            within each synchronization interval to achieve faster
```

```
          synchronization.";
        }
        leaf iburst {
          type boolean;
          default "false";
          description
            "Sends a series of packets instead of a single packet
            within the initial synchronization interval to achieve
            faster initial synchronization.";
        }
        leaf source {
          type if:interface-ref;
          description
            "The interface whose ip address this association used
            as source address.";
        }
        uses comman-attributes {
          description
            "Common attribute like port, version, min and max poll.";
        }
        leaf stratum {
          type ntp-stratum;
          config "false";
          description
            "Indicates the stratum of the reference clock.";
        }
        leaf refid {
          type union {
            type inet:ipv4-address;
            type binary {
              length "4";
            }
            type string {
              length "4";
            }
          }
          config "false";
          description
            "Reference clock type or address for the peer.";
        }
        leaf reach {
          type uint8;
          config "false";
          description
             "Indicates the reachability of the configured
             server or peer.";
        }
        leaf unreach {
```

```
          type uint8;
          config "false";
          description
            "Indicates the unreachability of the configured
            server or peer.";
        }
        leaf poll {
          type uint8;
          config "false";
          description
            "Indicates the polling interval for current,
            in seconds.";
        }
        leaf now {
          type uint32;
          config "false";
          description
            "Indicates the time since the NTP packet was
            not received or last synchronized, in seconds.";
        }
        leaf offset {
          type decimal64 {
            fraction-digits 4;
          }
          config "false";
          description
            "Indicates the offset between the local clock
            and the superior reference clock.";
        }
        leaf delay {
          type decimal64 {
            fraction-digits 2;
          }
          config "false";
          description
            "Indicates the delay between the local clock
            and the superior reference clock.";
        }
        leaf dispersion {
          type decimal64 {
            fraction-digits 2;
          }
          config "false";
          description
            "Indicates the dispersion between the local
            clock and the superior reference clock.";
        }
        leaf originate-time {
```

```
          type yang:date-and-time;
          config "false";
          description
            "Indicates packet originate timestamp(T1).";
        }
        leaf receive-time {
          type yang:date-and-time;
          config "false";
          description
            "Indicates packet receive timestamp(T2).";
        }
        leaf transmit-time {
          type yang:date-and-time;
          config "false";
          description
            "Indicates packet transmit timestamp(T3).";
        }
        leaf input-time {
          type yang:date-and-time;
          config "false";
          description
            "Indicates packet input timestamp(T4).";
        }
        container ntp-statistics {
          config "false";
          description
            "Per Peer packet send and receive statistic.";
          uses statistics {
              description
                "NTP send and receive packet statistic.";
          }
        }
      }

      container interface {
        description
          "Configuration parameters for NTP interfaces.";
        list interface {
          key "interface";
          description
            "List of interfaces.";
          leaf interface {
                type if:interface-ref;
                description
                  "The interface name.";
          }

          container broadcast-server {
```

```
            presence
              "NTP broadcast-server is configured";
            description
              "Configuration of broadcast server.";
            leaf ttl {
              type uint8;
              description
                "Specifies the time to live (TTL) of a
                broadcast packet.";
            }
            container authentication{
              description
                "Authentication type.";
                uses authentication-type-param;
            }
            uses comman-attributes {
              description
                "Common attribute like port, version, min and max poll.";
            }
          }

          container broadcast-client {
            presence
              "NTP broadcast-client is configured";
            description
              "Configuration of broadcast-client.";
          }

          list multicast-server {
            key "address";
            description
              "Configuration of multicast server.";
            leaf address {
               type rt-types:ip-multicast-group-address;
               description
                 "The IP address to send NTP multicast packets.";
            }
            leaf ttl {
              type uint8;
              description
                "Specifies the time to live (TTL) of a
                multicast packet.";
            }
            container authentication{
              description
                "Authentication type.";
                uses authentication-type-param;
            }
```

```
            uses comman-attributes {
              description
                "Common attribute like port, version, min and max poll.";
            }
          }
          list multicast-client {
            key "address";
            description
              "Configuration of multicast-client.";
            leaf address {
              type rt-types:ip-multicast-group-address;
              description
                "The IP address of the multicast group to join.";
            }
          }
          list manycast-server {
            key "address";
            description
                 "Configuration of manycast server.";
            leaf address {
              type rt-types:ip-multicast-group-address;
              description
                "The multicast group IP address to receive
                manycast client messages .";
            }
          }
          list manycast-client {
            key "address";
            description
              "Configuration of manycast-client.";
            leaf address {
              type rt-types:ip-multicast-group-address;
              description
                "The group IP address that the manycast client
                broadcasts the request message to.";
            }
            container authentication{
              description
                "Authentication type.";
                uses authentication-type-param;
            }
            leaf ttl {
              type uint8;
              description
                "Specifies the maximum time to live (TTL) for
                the expanding ring search.";
            }
            leaf minclock {
```

```
            type uint8;
            description
              "The minimum manycast survivors in this
              association.";
          }
          leaf maxclock {
            type uint8;
            description
              "The maximum manycast candidates in this
              association.";
          }
          leaf beacon {
            type uint8;
            description
              "The maximum interval between beacons in this
              association.";
          }
          uses comman-attributes {
            description
              "Common attribute like port, version, min and max poll.";
          }
        }
       }
      }
     }

     /* Operational state data */

     container ntp-statistics {
       config "false";
       description
         "Total NTP packet statistic.";
       uses statistics {
           description
             "NTP send and receive packet statistic.";
       }
     }
    }
    <CODE ENDS>
```

6. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

URI: urn:ietf:params:xml:ns:yang:ietf-ntp

Registrant Contact: The NETMOD WG of the IETF.

XML: N/A; the requested URI is an XML namespace.

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

Name: ietf-ntp

Namespace: urn:ietf:params:xml:ns:yang:ietf-ntp

Prefix: ntp

Reference: RFC XXXX

7. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

8. Acknowledgments

The authors would like to express their thanks to Sladjana Zoric, Danny Mayer, Harlan Stenn, Ulrich Windl and Miroslav Lichvar for their review and suggestions.

9. References

9.1. Normative References

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[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, <http://www.rfc-editor.org/info/rfc5905>.

[RFC5907] Gerstung, H., Elliott, C., and B. Haberman, Ed., "Definitions of Managed Objects for Network Time Protocol Version 4 (NTPv4)", RFC 5907, DOI 10.17487/RFC5907, June 2010, <http://www.rfc-editor.org/info/rfc5907>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

9.2. Informative References

[I-D.ietf-netmod-revised-datastores] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-02 (work in progress), May 2017.

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

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <http://www.rfc-editor.org/info/rfc7317>.

Authors' Addresses

Nan Wu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: eric.wu@huawei.com

Anil Kumar S N
RtBrick Inc.
Bangalore, Kanataka 560037
India

Email: anil.ietf@gmail.com

Yi Zhao
Ericsson
China Digital Kingdom Bld., No.1 WangJing North Rd.
Beijing 100102
China

Email: yi.z.zhao@ericsson.com

Dhruv Dhody
Huawei
Divyashree Techno Park, Whitefield
Bangalore, Kanataka 560066
India

Email: dhruv.ietf@gmail.com

Ankit kumar Sinha
Huawei
Divyashree Techno Park, Whitefield
Bangalore, Kanataka 560066
India

Email: ankit.ietf@gmail.com