

Audio/Video Transport Core
Maintenance
Internet-Draft
Intended status: Standards Track
Expires: September 26, 2014

A. Williams
Audinate
K. Gross
AVA Networks
R. van Brandenburg
H. Stokking
TNO
March 25, 2014

RTP Clock Source Signalling
draft-ietf-avtcore-clksrc-11

Abstract

NTP format timestamps are used by several RTP protocols for synchronisation and statistical measurements. This memo specifies SDP signalling identifying timestamp reference clock sources and SDP signalling identifying the media clock sources in a multimedia session.

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 September 26, 2014.

Copyright Notice

Copyright (c) 2014 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	4
2. Applications	4
3. Definitions	5
4. Timestamp Reference Clock Source Signalling	6
4.1. Clock synchronization	6
4.2. Identifying NTP Reference Clocks	7
4.3. Identifying PTP Reference Clocks	7
4.4. Identifying Global Reference Clocks	9
4.5. Private Reference Clocks	9
4.6. Local Reference Clocks	9
4.7. Traceable Reference Clocks	9
4.8. SDP Signalling of Timestamp Reference Clock Source	10
4.8.1. Examples	12
5. Media Clock Source Signalling	13
5.1. Asynchronously Generated Media Clock	13
5.2. Direct-Referenced Media Clock	13
5.3. Stream-Referenced Media Clock	15
5.4. SDP Signalling of Media Clock Source	16
5.5. Examples	18
6. Signalling Considerations	20
6.1. Usage in Offer/Answer	20
6.1.1. Indicating Support for Clock Source Signalling	21
6.1.2. Timestamp Reference Clock	21
6.1.3. Media Clock	21
6.2. Usage Outside of Offer/Answer	22
7. Security Considerations	22
8. IANA Considerations	23
8.1. Reference Clock SDP Parameter	23
8.2. Media Clock SDP Parameter	24
8.3. Timestamp Reference Clock Source Parameters Registry	24
8.4. Media Clock Source Parameters Registry	25
8.5. Source-level Attributes	26
8.5.1. Source-level Timestamp Reference Clock Attribute	26
8.5.2. Source-level Media Clock Attribute	26
9. Acknowledgements	26
10. References	26
10.1. Normative References	26
10.2. Informative References	28
Authors' Addresses	29

1. Introduction

RTP protocols use NTP format timestamps to facilitate multimedia session synchronisation and for providing estimates of round trip time (RTT) and other statistical parameters.

Information about media clock timing exchanged in NTP format timestamps may come from a clock which is synchronised to a global time reference, but this cannot be assumed nor is there a standardised mechanism available to indicate that timestamps are derived from a common reference clock. Therefore, RTP implementations typically assume that NTP timestamps are taken using unsynchronised clocks and must compensate for absolute time differences and rate differences. Without a shared reference clock, RTP can time align flows from the same source at a given receiver using relative timing, however tight synchronisation between two or more different receivers (possibly with different network paths) or between two or more senders is not possible.

High performance AV systems often use a reference media clock distributed to all devices in the system. The reference media clock is often distinct from the reference clock used to provide timestamps. A reference media clock may be provided along with an audio or video signal interface, or via a dedicated clock signal (e.g. genlock [SMPTE-318-1999] or audio word clock [AES11-2009]). If sending and receiving media clocks are known to be synchronised to a common reference clock, performance can be improved by minimising buffering and avoiding rate conversion.

This specification defines SDP signalling of timestamp reference clock sources and media reference clock sources.

2. Applications

Timestamp reference clock source and media clock signalling benefit applications requiring synchronised media capture or playout and low latency operation.

Examples include, but are not limited to:

Social TV : RTCP for inter-destination media synchronization
[I-D.ietf-avtcore-idms] defines social TV as the combination of media content consumption by two or more users at different devices and locations and real-time communication between those users. An example of Social TV, is where two or more users are watching the same television broadcast at different devices and/or locations, while communicating with each other using text, audio

and/or video. A skew in the media playout of the two or more users can have adverse effects on their experience. A well-known use case here is one friend experiencing a goal in a football match well before or after other friends.

Video Walls : A video wall consists of multiple computer monitors, video projectors, or television sets tiled together contiguously or overlapped in order to form one large screen. Each of the screens reproduces a portion of the larger picture. In some implementations, each screen or projector may be individually connected to the network and receive its portion of the overall image from a network-connected video server or video scaler. Screens are refreshed at 50 or 60 hertz or potentially faster. If the refresh is not synchronized, the effect of multiple screens acting as one is broken.

Networked Audio : Networked loudspeakers, amplifiers and analogue I/O devices transmitting or receiving audio signals via RTP can be connected to various parts of a building or campus network. Such situations can for example be found in large conference rooms, legislative chambers, classrooms (especially those supporting distance learning) and other large-scale environments such as stadiums. Since humans are more susceptible to differences in audio delay, this use case needs even more accuracy than the video wall use case. Depending on the exact application, the need for accuracy can then be in the range of microseconds [Olsen].

Sensor Arrays : Sensor arrays contain many synchronised measurement elements producing signals which are then combined to form an overall measurement. Accurate capture of the phase relationships between the various signals arriving at each element of the array is critically important for proper operation. Examples include towed or fixed sonar arrays, seismic arrays and phased arrays used in radar applications, for instance.

3. Definitions

The following definitions are used in this draft:

media level : Media level information applies to a single SDP media stream. In an SDP description, media-level information appears after each "m"-line.

multimedia session : A set of multimedia senders and receivers as well as the data streams flowing from senders to receivers. The Session Description Protocol (SDP) [RFC4566] describes multimedia sessions.

RTP media stream : A single stream of RTP packets identified by an RTP SSRC.

RTP media sender : The device generating an associated RTP media stream

SDP media stream : An RTP session potentially containing more than one RTP source. SDP media descriptions beginning with an "m"-line define the parameters of an SDP media stream.

session level : Session level information applies to an entire multimedia session. In an SDP description, session-level information appears before the first "m"-line.

source level : Source level information applies to a specific RTP media stream. Source-Specific Media Attributes in the Session Description Protocol (SDP) [RFC5576] defines how source-level information is included into an SDP session description.

traceable time : A clock is considered to provide traceable time if it can be proven to be synchronised to International Atomic Time (TAI). Coordinated Universal Time (UTC) is a time standard synchronized to TAI. UTC is therefore also considered traceable time once leap seconds have been taken into account. GPS [IS-GPS-200F] is commonly used to provide a TAI traceable time reference. Some network time synchronisation protocols (e.g. PTP [IEEE1588-2008], NTP) can explicitly indicate that the master clock is providing a traceable time reference over the network.

4. Timestamp Reference Clock Source Signalling

The NTP format timestamps used by RTP are taken by reading a local real-time clock at the sender or receiver. This local clock may be synchronised to another clock (time source) by some means or it may be unsynchronised. A variety of methods are available to synchronise local clocks to a reference time source, including network time protocols (e.g. NTP [RFC5905], PTP [IEEE1588-2008]) and radio clocks (e.g. GPS [IS-GPS-200F]).

The following sections describe and define SDP signalling, indicating whether and how the local timestamping clock in an RTP sender/receiver is synchronised to a reference clock.

4.1. Clock synchronization

Two or more local clocks that are sufficiently synchronised will produce timestamps for a given RTP event can be used as if they came

from the same clock. Providing they are sufficiently synchronised, timestamps produced in one RTP sender or receiver can be directly compared to a local clock in another RTP sender or receiver.

The accuracy of synchronisation required is application dependent. See Applications (Section 2) section for a discussion of applications and their corresponding requirements. To serve as a reference clock, clocks must minimally be syntonized (exactly frequency matched) to one another.

Sufficient synchronisation can typically be achieving by using a network time protocol (e.g. NTP, 802.1AS, IEEE 1588-2008) to synchronize all devices to a single master clock.

Another approach is to use clocks providing a global time reference (e.g. GPS, Galileo, GLONASS). This concept may be used in conjunction with network time protocols as some protocols (e.g. PTP, NTP) allow master clocks to indicate explicitly that they are providing traceable time.

4.2. Identifying NTP Reference Clocks

A single NTP server is identified by hostname (or IP address) and an optional port number. If the port number is not indicated, it is assumed to be the standard NTP port (123).

Two or more NTP servers MAY be listed at the same level in the session description to indicate that all of the listed servers deliver the same reference time and may be used interchangeably. RTP senders and receivers are assured proper synchronization regardless of which server they choose and, in support of fault tolerance, may switch servers while streaming.

4.3. Identifying PTP Reference Clocks

The IEEE 1588 Precision Time Protocol (PTP) family of clock synchronisation protocols provides a shared reference clock in an network - typically a LAN. IEEE 1588 provides sub-microsecond synchronisation between devices on a LAN and typically locks within seconds at startup. With support from Ethernet switches, IEEE 1588 protocols can achieve nanosecond timing accuracy in LANs. Network interface chips and cards supporting hardware time-stamping of timing critical protocol messages are also available.

Three flavours of IEEE 1588 are in use today:

- o IEEE 1588-2002 [IEEE1588-2002]: the original "Standard for a Precision Clock Synchronization Protocol for Networked Measurement

and Control Systems". This is also known as IEEE1588v1 or PTPv1.

- o IEEE 1588-2008 [IEEE1588-2008]: the second version of the "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". This is a revised version of the original IEEE1588-2002 standard and is also known as IEEE1588v2 or PTPv2. IEEE 1588-2008 is not protocol compatible with IEEE 1588-2002.
- o IEEE 802.1AS [IEEE802.1AS-2011]: "Timing and Synchronization for Time Sensitive Applications in Bridged Local Area Networks". This is a Layer-2 only profile of IEEE 1588-2008 for use in Audio/Video Bridged LANs as described in IEEE 802.1BA-2011 [IEEE802.1BA-2011].

Each IEEE 1588 clock is identified by an EUI-64 called a "ClockIdentity". A slave clock using one of the IEEE 1588 family of network time protocols acquires the ClockIdentity/EUI-64 of the grandmaster clock that is the ultimate source of timing information for the network. A boundary clock which is itself slaved to another boundary clock or the grandmaster passes the grandmaster ClockIdentity through to its slaves.

Several instances of the IEEE 1588 protocol may operate independently on a single network, forming distinct PTP domains, each of which may have a different grandmaster clock. As the IEEE 1588 standards have developed, the definition of PTP domains has changed. IEEE 1588-2002 identifies protocol subdomains by a textual name, but IEEE 1588-2008 identifies protocol domains using a numeric domain number. 802.1AS is a Layer-2 profile of IEEE 1588-2008 supporting a single numeric clock domain (0).

When PTP domains are signalled via SDP, senders and receivers SHOULD check that both grandmaster ClockIdentity and PTP domain match when determining clock equivalence.

Two or more IEEE 1588 clocks MAY be listed at the same level in the session description to indicate that all of the listed clocks are candidate grandmaster clocks for the domain or deliver the same reference time and may be used interchangeably. RTP senders and receivers are assured proper synchronization regardless of which synchronization source they choose and, in support of fault tolerance, may switch reference clock source while streaming.

The PTP protocols employ a distributed election protocol called the "Best Master Clock Algorithm" (BMCA) to determine the active clock master. The clock master choices available to BMCA can be restricted or biased by configuration parameters to influence the election process. In some systems it may be desirable to limit the number of

possible PTP clock masters to avoid the need to re-signal timestamp reference clock sources when the clock master changes.

4.4. Identifying Global Reference Clocks

Global reference clocks provide a source of traceable time, typically via a hardware radio receiver interface. Examples include GPS, Galileo and GLONASS. Apart from the name of the reference clock system, no further identification is required.

4.5. Private Reference Clocks

In other systems, all RTP senders and receivers may use a timestamp reference clock that is not provided by one of the methods listed above. Examples may include the reference time information provided by digital television or cellular services. These sources are identified as "private" reference clocks. All RTP senders and receivers in a session using a private reference clock are assumed to have a mechanism outside this specification for determining whether their timestamp reference clocks are equivalent.

4.6. Local Reference Clocks

RFC 3550 allows senders and receivers to either use a local wall clock reference for their NTP timestamps or, by setting the timestamp field to 0, to supply no timestamps at all. Both are common practice in embedded RTP implementations. These clocks are identified as "local" and can only be assumed to be equivalent to clocks originating from the same device.

4.7. Traceable Reference Clocks

A timestamp reference clock source may be labelled "traceable" if it is known to be delivering traceable time. Providing adjustments are made for differing epochs, timezones and leap seconds, timestamps taken using clocks synchronised to a traceable time source can be directly compared even if the clocks are synchronised to different sources or via different mechanisms.

Marking a clock as traceable allows additional information (e.g. IP addresses, PTP master identifiers and the like) to be omitted from the SDP since any traceable clock available at the answerer is considered to be an appropriate timestamp reference clock. For example, an offerer could specify `ts-refclk:ntp=/traceable/` and the answerer could use GPS as a reference clock since GPS is a source of traceable time.

4.8. SDP Signalling of Timestamp Reference Clock Source

Specification of the timestamp reference clock source may be at any or all levels (session, media or source) of an SDP description (see level definitions (Section 3) earlier in this document for more information).

Timestamp reference clock source signalling included at session-level provides default parameters for all RTP sessions and sources in the session description. More specific signalling included at the media level overrides default session level signalling. More specific signalling included at the source level overrides default media level signalling.

If timestamp reference clock source signalling is included anywhere in an SDP description, it must be properly defined for all levels in the description. This may simply be achieved by providing default signalling at the session level.

Timestamp reference clock parameters may be repeated at a given level (i.e. for a session or source) to provide information about additional servers or clock sources. If the attribute is repeated at a given level, all clocks described at that level are assumed to be equivalent. Traceable time sources **MUST NOT** be mixed with non-traceable time sources at any given level.

Note that clock source parameters may change from time to time, for example, as a result of a PTP clock master election. The SIP [RFC3261] protocol supports re-signalling of updated SDP information, however other protocols may require additional notification mechanisms.

General forms of usage:

session level: a=ts-refclk:<clksrc>

media level: a=ts-refclk:<clksrc>

source level: a=ssrc:<ssrc-id> ts-refclk:<clksrc>

ABNF [RFC5234] grammar for the timestamp reference clock attribute:

; external references:

POS-DIGIT = <See RFC 4566>
token = <See RFC 4566>
byte-string = <See RFC 4566>
DIGIT = <See RFC 5324>
HEXDIG = <See RFC 5324>
CRLF = <See RFC 5324>

```

hostport      = <See RFC 3261, with revisions from RFC 5954>

timestamp-refclk = "ts-refclk:" clksrc CRLF

clksrc = ntp / ptp / gps / gal / glonass / local / private / clksrc-ext

clksrc-ext      = clksrc-param-name clksrc-param-value
clksrc-param-name = token
clksrc-param-value = ["=" byte-string ]

ntp              = "ntp=" ntp-server-addr
ntp-server-addr = hostport / "/traceable/"

ptp              = "ptp=" ptp-version ":" ptp-server
ptp-version      = "IEEE1588-2002"
                  / "IEEE1588-2008"
                  / "IEEE802.1AS-2011"
                  / ptp-version-ext
ptp-version-ext  = token

ptp-server       = ptp-gmid [ ":" ptp-domain ]
                  / "traceable"
ptp-gmid         = EUI64
ptp-domain       = ptp-domain-name / ptp-domain-nmbr

; PTP domain allowed characters: 0x21-0x7E (IEEE 1588-2002)
ptp-domain-name = "domain-name=" 1*16ptp-domain-char
ptp-domain-char = %x21-7E

; PTP domain allowed number range: 0-127 (IEEE 1588-2008)
ptp-domain-nmbr = "domain-nmbr=" ptp-domain-dgts
ptp-domain-dgts = ptp-domain-n1 / ptp-domain-n2 / ptp-domain-n3
ptp-domain-n1   = DIGIT           ; 0-9
ptp-domain-n2   = POS-DIGIT DIGIT ; 10-99
ptp-domain-n3   = ("10"/"11") DIGIT ; 100-119
                  / "12" %x30-37    ; 120-127

gps      = "gps"
gal      = "gal"
glonass  = "glonass"
local    = "local"
private  = "private" [ ":"traceable" ]

EUI64 = 7(2HEXDIG "-") 2HEXDIG

```

Figure 1: Timestamp Reference Clock Source Signalling

4.8.1. Examples

Figure 2 shows an example SDP description with a timestamp reference clock source defined at the session level.

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 192.0.2.1
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 233.252.0.1/64
t=2873397496 2873404696
a=recvonly
a=ts-refclk:ntp=/traceable/
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
```

Figure 2: Timestamp reference clock definition at the session level

Figure 3 shows an example SDP description with timestamp reference clock definitions at the media level overriding the session level defaults.

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 192.0.2.1
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 233.252.0.1/64
t=2873397496 2873404696
a=recvonly
a=ts-refclk:local
m=audio 49170 RTP/AVP 0
a=ts-refclk:ntp=203.0.113.10
a=ts-refclk:ntp=198.51.100.22
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
a=ts-refclk:ptp=IEEE802.1AS-2011:39-A7-94-FF-FE-07-CB-D0
```

Figure 3: Timestamp reference clock definition at the media level

Figure 4 shows an example SDP description with a timestamp reference clock definition at the source level overriding the session level default.

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 192.0.2.1
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 233.252.0.1/64
t=2873397496 2873404696
a=recvonly
a=ts-refclk:local
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
a=ssrc:12345 ts-refclk:ptp=IEEE802.1AS-2011:39-A7-94-FF-FE-07-CB-D0
```

Figure 4: Timestamp reference clock signalling at the source level

5. Media Clock Source Signalling

The media clock source for a stream determines the timebase used to advance the RTP timestamps included in RTP packets. The media clock may be asynchronously generated by the sender, it may be generated in fixed relationship to the reference clock or it may be generated with respect to another stream on the network (which is presumably being received by the sender).

5.1. Asynchronously Generated Media Clock

In the simplest sender implementation, the sender generates media by sampling audio or video according to a free-running local clock. The RTP timestamps in media packets are advanced according to this media clock and packet transmission is typically timed to regular intervals on this timeline. The sender may or may not include an NTP timestamp in sender reports to allow mapping of this asynchronous media clock to a reference clock.

The asynchronously generated media clock is the assumed mode of operation when there is no signalling of media clock source. Alternatively, asynchronous media clock may be explicitly signalled.

```
a=mediaclock:sender
```

5.2. Direct-Referenced Media Clock

A media clock may be directly derived from a reference clock. For this case it is required that a reference clock be specified with an `a=ts-refclk` attribute (Section 4.8).

The signalling optionally indicates a media clock offset value. The offset indicates the RTP timestamp value at the epoch (time of origin) of the reference clock. To use the offset, implementations need to compute RTP timestamps from reference clocks. To simplify these calculations, streams utilizing offset signalling SHOULD use a TAI timestamp reference clock to avoid complications introduced by leap seconds. See [I-D.ietf-avtcore-leap-second] for further discussion of leap-second issues in timestamp reference clocks.

To compute the RTP timestamp against an IEEE 1588 (TAI-based) reference, the time elapsed between the 00:00:00 1 January 1970 IEEE 1588 epoch and the current time must be computed. Between the epoch and 1 January 2013, there were 15,706 days (including extra days during leap years). Since there are no leap seconds in a TAI reference, there are exactly 86,400 seconds during each of these days or a total of 1,356,998,400 seconds from the epoch to 00:00:00 1 January 2013. A 90 kHz RTP clock for a video stream would have advanced 122,129,856,000,000 units over this period. With a signalled offset of 0, the RTP clock value modulo the 32-bit unsigned representation in the RTP header would have been 2,460,938,240 at 00:00:00 1 January 2013. If an offset of 23,465 had been signalled, the clock value would have been 2,460,961,705.

In order to use an NTP reference, the actual time elapsed between the 00:00:00, 1 January 1900 NTP epoch to the current time must be computed. 2,208,988,800 seconds elapsed between the NTP epoch and 00:00:00 1 January 1970 [RFC0868]. Between the beginning of 1970 and 2013, there were 15,706 days elapsed (including extra days during leap years) and 25 leap seconds inserted. There is therefore a total of 3,565,987,225 seconds from the NTP epoch to 00:00:00 1 January 2013. A 90 kHz RTP clock for a video stream would have advanced 320,938,850,250,000 units over this period. With a signalled offset of 0, the RTP clock value modulo the 32-bit unsigned representation would have been 1,714,023,696 at 00:00:00 1 January 2013.

If no offset is signalled, the offset can be inferred at the receiver by examining RTCP sender reports which contain NTP and RTP timestamps which combined define a mapping. The NTP/RTP timestamp mapping provided by RTCP SRs takes precedence over that signalled through SDP, however the media clock rate implied by the SRs MUST be consistent with the rate signalled.

A rate modifier may be specified. The modifier is expressed as the ratio of two integers and modifies the rate specified or implied by the media description by this ratio. If omitted, the rate is assumed to be the exact rate specified or implied by the media format. For example, without a rate specification, the RTP clock for an 8 kHz G.711 audio stream will advance exactly 8000 units for each second

advance in the reference clock from which it is derived.

The rate modifier is primarily useful for accommodating certain "oddball" audio sample rates associated with NTSC video (see Figure 7). Modified rates are not advised for video streams which generally use a 90 kHz RTP clock regardless of frame rate or sample rate used for embedded audio.

```
a=mediaclock:direct[=<offset>] [rate=<rate numerator>/<rate
denominator>]
```

5.3. Stream-Referenced Media Clock

A common synchronisation architecture for audio/visual systems involves distributing a reference media clock from a master device to a number of slave devices, typically by means of a cable. Examples include audio word clock distribution and video black burst distribution. In this case, the media clock is locally generated, often by a crystal oscillator and is not locked to a timestamp reference clock.

To support this architecture across a network, a master clock identifier is associated with an RTP media stream carrying media clock timing information from a master device. The master clock identifier represents a media clock source in the master device. Slave devices in turn associate the master media clock identifier with streams they transmit, signalling the synchronisation relationship between the master and the transmitter's media clock.

Slave devices recover media clock timing from the clock master stream, using it to synchronise the slave media clock with the master. Timestamps in the master clock RTP media stream are taken using the timestamp reference clock shared by the master and slave devices. The timestamps communicate information about media clock timing (rate, phase) from the master to the slave devices. Timestamps are communicated in the usual RTP fashion via RTCP SRs, or via the RFC6051 [RFC6051] header extension. The stream media format may indicate other clock information, such as the nominal rate.

Note that slaving of a device media clock to a master device does not affect the usual RTP lip sync / time alignment algorithms. Time aligned playout of two or more RTP sources still relies upon NTP timestamps supplied via RTCP SRs or by the RFC6051 timestamp header extension.

In a given system, master clock identifiers must uniquely identify a single media clock source. Such identifiers MAY be manually configured, however identifiers SHOULD be generated according to the

"short-term persistent RTCP CNAME" algorithm as described in RFC7022 [RFC7022]. Master clock identifiers not already in base64 format MUST be encoded as a base64 strings when used in SDP. Although the RTCP CNAME algorithm is used to generate the master clock identifier, it is used to tag RTP sources in SDP descriptions and does not appear in RTCP as a CNAME.

A reference stream can be an RTP stream or AVB stream based on the IEEE 1722 [IEEE1722] standard.

An RTP clock master stream SHOULD be identified at the source level by an SSRC [RFC5576] and master clock identifier. An RTP stream that provides media clock timing directly from a reference media clock (e.g. internal crystal, audio word clock or video blackburst signal) SHOULD tag the stream as a master clock source using the "src:" prefix. If master clock identifiers are declared at the media or session level, all RTP sources at or below the level of declaration MUST provide equivalent timing to a slave receiver.

```
a=ssrc:<ssrc> mediack:id=src:<media-clktag> sender
```

```
a=mediack:id=src:<media-clktag> sender
```

A transmitted RTP stream slaved to media clock master is signalled by including master clock identifier:

```
a=mediack:id=<media-clktag> sender
```

An RTP media sender indicates that it is slaved to an IEEE 1722 clock master via a stream identifier (an EUI-64):

```
a=mediack:IEEE1722=<StreamID>
```

An RTP media sender may gateway IEEE 1722 media clock timing to RTP:

```
a=mediack:id=src:<media-clktag> IEEE1722=<StreamID>
```

5.4. SDP Signalling of Media Clock Source

Specification of the media clock source may be at any or all levels (session, media or source) of an SDP description (see level definitions (Section 3) earlier in this document for more information).

Media clock source signalling included at session level provides default parameters for all RTP sessions and sources in the session description. More specific signalling included at the media level overrides default session level signalling. Further, source-level

signalling overrides media clock source signalling at the enclosing media level and session level.

Media clock source signalling may be present or absent on a per-stream basis. In the absence of media clock source signals, receivers assume an asynchronous media clock generated by the sender.

Media clock source parameters may be repeated at a given level (i.e. for a session or source) to provide information about additional clock sources. If the attribute is repeated at a given level, all clocks described at that level are comparable clock sources and may be used interchangeably.

General forms of usage:

session level: a=mediaclock:<mediaclock>

media level: a=mediaclock:<mediaclock>

source level: a=ssrc:<ssrc-id> mediaclock:<mediaclock>

```

ABNF [RFC5234] grammar for the media clock reference attribute:
; external references:
integer      = <See RFC 4566>
token       = <See RFC 4566>
byte-string  = <See RFC 4566>
base64      = <See RFC 4566>
SP          = <See RFC 5234>
DIGIT       = <See RFC 5234>
HEXDIG      = <See RFC 5234>

media-clksrc = "mediaclk:" [media-clkid SP] mediaclock

media-clkid  = "id=" [ "src:" ] media-clktag
media-clktag = base64

mediaclock   = sender / direct / ieee1722-streamid / mediaclock-ext

mediaclock-ext      = mediaclock-param-name mediaclock-param-value
mediaclock-param-name = token
mediaclock-param-value = [ "=" byte-string ]

sender = "sender"
direct = "direct" [ "=" 1*DIGIT ] [SP rate]
rate   = "rate=" integer "/" integer

ieee1722-streamid = "IEEE1722=" avb-stream-id
avb-stream-id     = EUI64
EUI64 = 7(2HEXDIG "-") 2HEXDIG

```

Figure 5: Media Clock Source Signalling

5.5. Examples

Figure 6 shows an example SDP description 8 channels of 24-bit, 48 kHz audio transmitted as a multicast stream. Media clock is derived directly from an IEEE 1588-2008 reference.

```
v=0
o=- 1311738121 1311738121 IN IP4 192.0.2.1
c=IN IP4 233.252.0.1/64
s=
t=0 0
m=audio 5004 RTP/AVP 96
a=rtpmap:96 L24/48000/8
a=sendonly
a=ts-refclk:ptp=IEEE1588-2008:39-A7-94-FF-FE-07-CB-D0:0
a=mediaclock:direct=963214424
```

Figure 6: Media clock directly referenced to IEEE 1588-2008

Figure 7 shows an example SDP description 2 channels of 24-bit, 44056 kHz NTSC "pull-down" media clock derived directly from an IEEE 1588-2008 reference clock

```
v=0
o=- 1311738121 1311738121 IN IP4 192.0.2.1
c=IN IP4 233.252.0.1/64
s=
t=0 0
m=audio 5004 RTP/AVP 96
a=rtpmap:96 L24/44100/2
a=sendonly
a=ts-refclk:ptp=IEEE1588-2008:39-A7-94-FF-FE-07-CB-D0:0
a=mediaclock:direct=963214424 rate=1000/1001
```

Figure 7: "Oddball" sample rate directly referenced to IEEE 1588-2008

Figure 8 shows the same 48 kHz audio transmission from Figure 6 with media clock derived from another RTP stream.

```
v=0
o=- 1311738121 1311738121 IN IP4 192.0.2.1
c=IN IP4 233.252.0.1/64
s=
t=0 0
m=audio 5004 RTP/AVP 96
a=rtpmap:96 L24/48000/2
a=sendonly
a=ts-refclk:ptp=IEEE1588-2008:39-A7-94-FF-FE-07-CB-D0:0
a=mediaclock:id=MDA6NjA6MmI6MjA6MTI6MWY= sender
```

Figure 8: RTP stream with media clock slaved to a master

Figure 9 shows the same 48 kHz audio transmission from Figure 6 with media clock derived from an IEEE 1722 AVB stream.

```
v=0
o=- 1311738121 1311738121 IN IP4 192.0.2.1
c=IN IP4 233.252.0.1/64
s=
t=0 0
m=audio 5004 RTP/AVP 96
a=rtpmap:96 L24/48000/2
a=sendonly
a=ts-refclk:ptp=IEEE1588-2008:39-A7-94-FF-FE-07-CB-D0:0
a=mediaclock:IEEE1722=38-D6-6D-8E-D2-78-13-2F
```

Figure 9: RTP stream with media clock slaved to an IEEE1722 master device

6. Signalling Considerations

Signalling of timestamp reference clock source (Section 4.8) and media clock source (Section 5.4) is defined to be used either by applications that implement the SDP Offer/Answer model [RFC3264] or by applications that use SDP to describe media and transport configurations.

A description SHOULD include both reference clock signalling and media clock signalling. If no reference clock is available, this SHOULD be signalled as a local reference (Section 4.6).

When no media clock signalling is present, an asynchronous media clock (Section 5.1) MUST be assumed. When no reference clock signalling is present, a local reference clock (Section 4.6) MUST be assumed.

If a reference clock is not signalled or a local reference is specified, the corresponding media clock may be established as rate synchronised with no assurance of time synchronisation.

When the description signals a direct-referenced media clock (Section 5.2), reference clock signalling is REQUIRED. Asynchronous and stream-referenced media clocks (Section 5.3) MAY be specified with or without a reference clock signalling.

6.1. Usage in Offer/Answer

During offer/answer, clock source signalling via SDP uses a declarative model. Supported media and/or reference clocks are specified in the offered SDP description. The answerer may accept or reject the offer in an application-specific way depending on the clocks that are available and the clocks that are offered. For

example, an answerer may choose to accept an offer that lacks a common clock by falling back to a lower performance mode of operation (e.g. by assuming reference or media clocks are local rather than shared). Conversely, the answerer may choose to reject the offer when the offered clock specifications indicate that the available reference and/or media clocks are incompatible.

While negotiation of reference clock and media clock attributes is not defined in this document, negotiation MAY be accomplished using the capabilities negotiation procedures defined in [RFC5939].

6.1.1.1. Indicating Support for Clock Source Signalling

An offerer or answerer indicates support for media clock signalling by including a reference or media clock specification in the SDP description. An offerer or answerer without specific reference or media clocks to signal SHOULD indicate support for clock source signalling by including a local reference clock (Section 4.6) specification in the SDP description.

6.1.1.2. Timestamp Reference Clock

If one or more of the reference clocks specified in the offer are usable by the answerer, the answerer SHOULD respond with an answer containing the subset of reference clock specifications in the offer that are usable by the answerer. If the answerer rejects the offer because the available reference clocks are incompatible, the rejection MUST contain at least one timestamp reference clock specification usable by the answerer so that appropriate information is available for debugging. If no external reference clock is available to the answerer a local reference clock (Section 4.6) specification SHOULD be included in the rejection.

In both offers and answers, multiple reference clock specifications indicate equivalent clocks from different sources which may be used interchangeably. RTP senders and receivers are assured proper synchronization regardless of which of the specified sources is chosen and, in support of fault tolerance, may switch clock sources while streaming.

6.1.1.3. Media Clock

If the media clock mode specified in the offer is acceptable to the answerer, the answerer SHOULD respond with an answer containing the same media clock specification as the offer. If the answerer rejects the offer because the available reference clocks are incompatible, the rejection MUST contain a media clock specification supported by the answerer so that appropriate information is available for

debugging. If no shared media clocks are available to the answerer an asynchronous media clock (Section 5.1) specification SHOULD be included in the rejection.

6.2. Usage Outside of Offer/Answer

SDP can be employed outside of the Offer/Answer context, for instance for multimedia sessions that are announced through the Session Announcement Protocol (SAP) [RFC2974], or streamed through the Real Time Streaming Protocol (RTSP) [RFC2326].

Devices using published descriptions to join sessions SHOULD assess their synchronization compatibility with the described session based on the clock source signalling and SHOULD NOT attempt to join a session with incompatible reference or media clocks.

7. Security Considerations

Entities receiving and acting upon an SDP message should note that a session description cannot be trusted unless it has been obtained by an authenticated transport protocol from a known and trusted source. Many different transport protocols may be used to distribute session description, and the nature of the authentication will differ from transport to transport. For some transports, security features are often not deployed. In case a session description has not been obtained in a trusted manner, the endpoint should exercise care because, among other attacks, the media sessions received may not be the intended ones, the destination where media is sent to may not be the expected one, any of the parameters of the session may be incorrect.

Incorrect reference or media clock parameters may cause devices or streams to synchronize to unintended clock sources. Normally this simply results in failure to establish a session or failure to synchronize once connected. Enough devices fraudulently assigned to a specific clock source (e.g. a particular IEEE 1588 grandmaster) may, however, constitute a successful denial of service attack on that source. Devices MAY wish to validate the integrity of the clock description through some means before connecting to unfamiliar clock sources.

The timestamp reference clocks negotiated by this protocol are used to provide media timing information to RTP. Negotiated timestamp reference clocks should not be relied upon to provide a secure time reference for security critical operations (e.g. the expiration of public key certificates).

8. IANA Considerations

This document defines two new SDP attributes: 'ts-refclk' and 'mediaclock', within the existing Internet Assigned Numbers Authority (IANA) registry of SDP Parameters.

This document also defines a new IANA registry subordinate to the IANA SDP Parameters registry: the Media Clock Source Parameters Registry. Within this new registry, this document defines an initial set of three media clock source parameters. Further, this document defines a second new IANA registry subordinate to the IANA SDP Parameters registry: the Timestamp Reference Clock Source Parameters Registry. Within this new registry, this document defines an initial six parameters.

8.1. Reference Clock SDP Parameter

The SDP attribute "ts-refclk" defined by this document is registered with the IANA registry of SDP Parameters as follows:

SDP Attributes ("att-field (both session and media level)" &
"att-field (source level)"):

Attribute name:	ts-refclk
Long form:	Timestamp reference clock source
Type of name:	att-field
Type of attribute:	Session, media and source level
Subject to charset:	No
Purpose:	See section 4 of this document
Reference:	This document
Values:	See section 8.3 of this document

Figure 10

The attribute has an extensible parameter field and therefore a registry for these parameters is required. This new registry is defined in Section 8.3.

8.2. Media Clock SDP Parameter

The SDP attribute "mediaclock" defined by this document is registered with the IANA registry of SDP Parameters as follows:

SDP Attributes ("att-field (both session and media level)" & "att-field (source level)"):

Attribute name:	mediaclock
Long form:	Media clock source
Type of name:	att-field
Type of attribute:	Session, media and source level
Subject to charset:	No
Purpose:	See section 5 of this document
Reference:	This document
Values:	See section 8.4 of this document

Figure 11

The attribute has an extensible parameter field and therefore a registry for these parameters is required. The new registry is defined in Section 8.4.

8.3. Timestamp Reference Clock Source Parameters Registry

This document creates a new IANA sub-registry called the Timestamp Reference Clock Source Parameters Registry, subordinate to the IANA SDP Parameters registry. Each entry in the Timestamp Reference Clock Source Parameters Registry contains:

Name:	Token used in the SDP description (clksrc-param-name)
Long name:	Descriptive name for the timestamp reference clock source
Reference:	Reference to the document describing the SDP token (clksrc-param-name) and syntax for the optional value associated with the token (mediaclock-param-value)

Initial values for the Timestamp Reference Clock Source Parameters registry are given below.

Future assignments are to be made through the Specification Required policy [RFC5226]. The Name field in the table corresponds to a new value corresponding to clksrc-param-name. The Reference must specify a syntax corresponding to clksrc-param-value.

Name	Long Name	Reference
ntp	Network Time Protocol	This document, section 4
ptp	Precision Time Protocol	This document, section 4
gps	Global Position System	This document, section 4
gal	Galileo	This document, section 4
glonass	Global Navigation Satellite System	This document, section 4
local	Local Clock	This document, section 4
private	Private Clock	This document, section 4

8.4. Media Clock Source Parameters Registry

This document creates a new IANA sub-registry called the Media Clock Source Parameters registry, subordinate to the IANA SDP Parameters registry. Each entry in the Media Clock Source Parameters Registry contains:

Name: Token used in the SDP description (mediaclock-param-name)

Long name: Descriptive name for the media clock source type

Reference: Reference to the document describing the SDP token (mediaclock-param-name) and syntax for the optional value associated with the token (mediaclock-param-value)

Initial values for the Media Clock Source Parameters registry are given below.

Future assignments are to be made through the Specification Required policy [RFC5226]. The Name field in the table corresponds to a new value corresponding to mediaclock-param-name. The Reference must specify a syntax corresponding to mediaclock-param-value.

Name	Long Name	Reference
sender	Asynchronously Generated Media Clock	This document, section 5
direct	Direct-Referenced Media Clock	This document, section 5
IEEE1722	IEEE1722 Media Stream Identifier	This document, section 5

8.5. Source-level Attributes

[RFC5576] requires new source-level attributes to be registered with the IANA registry named "att-field (source level)".

8.5.1. Source-level Timestamp Reference Clock Attribute

The source-level SDP attribute "ts-refclk" defined by this document is registered with the "att-field (source level)" IANA registry of SDP Parameters according to Figure 10.

8.5.2. Source-level Media Clock Attribute

The source-level SDP attribute "mediaclk" defined by this document is registered with the "att-field (source level)" IANA registry of SDP Parameters according to Figure 11.

9. Acknowledgements

The authors would like to thank Magnus Westerlund and Paul Kyzivat for valuable comments which resulted in important improvements to this document.

10. References

10.1. Normative References

[IEEE1588-2002]
Institute of Electrical and Electronics Engineers, "1588-2002 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2002, 2002, <<http://standards.ieee.org/findstds/standard/1588-2002.html>>.

[IEEE1588-2008]

Institute of Electrical and Electronics Engineers, "1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <<http://standards.ieee.org/findstds/standard/1588-2008.html>>.

[IEEE1722]

Institute of Electrical and Electronics Engineers, "IEEE Standard for Layer 2 Transport Protocol for Time Sensitive Applications in a Bridged Local Area Network", <<http://standards.ieee.org/findstds/standard/1722-2011.html>>.

[IEEE802.1AS-2011]

Institute of Electrical and Electronics Engineers, "Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks", <<http://standards.ieee.org/findstds/standard/802.1AS-2011.html>>.

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

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, June 2009.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, November 2010.

[RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla, "Guidelines for Choosing RTP Control Protocol (RTCP)

Canonical Names (CNAMEs)", RFC 7022, September 2013.

10.2. Informative References

[AES11-2009]

Audio Engineering Society, "AES11-2009: AES recommended practice for digital audio engineering - Synchronization of digital audio equipment in studio operations", <<http://www.aes.org/standards/>>.

[I-D.ietf-avtcore-idms]

Brandenburg, R., Stokking, H., Deventer, O., Boronat, F., Montagud, M., and K. Gross, "Inter-destination Media Synchronization using the RTP Control Protocol (RTCP)", draft-ietf-avtcore-idms-13 (work in progress), August 2013.

[I-D.ietf-avtcore-leap-second]

Gross, K. and R. Brandenburg, "RTP and Leap Seconds", draft-ietf-avtcore-leap-second-07 (work in progress), December 2013.

[IEEE802.1BA-2011]

Institute of Electrical and Electronics Engineers, "Audio Video Bridging (AVB) Systems", <<http://standards.ieee.org/findstds/standard/802.1BA-2011.html>>.

[IS-GPS-200F]

Global Positioning Systems Directorate, "Navstar GPS Space Segment/Navigation User Segment Interfaces", September 2011.

[Olsen]

Olsen, D., "Time Accuracy Requirements in Audio Networks", April 2007, <<http://www.ieee802.org/1/files/public/docs2007/as-dolsen-time-accuracy-0407.pdf>>.

[RFC0868]

Postel, J. and K. Harrenstien, "Time Protocol", STD 26, RFC 868, May 1983.

[RFC2326]

Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[RFC2974]

Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

[RFC3261]

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261,

June 2002.

[RFC5939] Andreassen, F., "Session Description Protocol (SDP) Capability Negotiation", RFC 5939, September 2010.

[SMPTE-318-1999]
Society of Motion Picture & Television Engineers,
"Television and Audio - Synchronization of 59.94- or 50-Hz
Related Video and Audio Systems in Analog and Digital
Areas - Reference Signals", <<http://standards.smpste.org/>>.

Authors' Addresses

Aidan Williams
Audinate
Level 1, 458 Wattle St
Ultimo, NSW 2007
Australia

Phone: +61 2 8090 1000
Fax: +61 2 8090 1001
Email: aidan.williams@audinate.com
URI: <http://www.audinate.com/>

Kevin Gross
AVA Networks
Boulder, CO
US

Email: kevin.gross@avanw.com
URI: <http://www.avanw.com/>

Ray van Brandenburg
TNO
Brassersplein 2
Delft 2612CT
the Netherlands

Phone: +31-88-866-7000
Email: ray.vanbrandenburg@tno.nl

Hans Stokking
TNO
Brassersplein 2
Delft 2612CT
the Netherlands

Email: hans.stokking@tno.nl

TICTOC Working Group
Internet-Draft
Intended status: Experimental
Expires: April 18, 2016

S. Davari
A. Oren
Broadcom Corp.
M. Bhatia
P. Roberts
Alcatel-Lucent
L. Montini
Cisco Systems
October 16, 2015

Transporting Timing messages over MPLS Networks
draft-ietf-tictoc-1588overmpls-07

Abstract

This document defines a method for transporting timing messages, such as Precision Time Protocol (PTP) or Network Time Protocol (NTP), over a Multiprotocol Label Switched (MPLS) network. The method facilitates efficient recognition of timing packets to enable their port level processing in both Label Edge Routers (LERs) and Label Switched Routers (LSRs).

The basic mechanism is to transport timing messages inside "Timing LSPs", which are dedicated MPLS Label Switched Paths (LSPs) that carry only timing, and possibly related Operations, Administration and Maintenance (OAM) or management packets, but do not carry customer traffic.

Two encapsulations methods are defined. The first transports UDP/IP encapsulated timing messages directly over the dedicated LSP. The second transports Ethernet encapsulated timing messages inside an Ethernet pseudowire.

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 18, 2016.

Copyright Notice

Copyright (c) 2015 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	3
2. Terminology	4
3. Problem Statement	5
4. Timing over MPLS Architecture	5
5. Dedicated LSPs for Timing messages	7
6. Timing over LSP Encapsulation	8
6.1. Timing over UDP/IP over MPLS Encapsulation	8
6.2. Timing over PW Encapsulation	8
7. Timing message Processing	9
8. Protection and Redundancy	10
9. ECMP and Entropy	10
10. PHP	11
11. OAM, Control and Management	11
12. QoS Considerations	11
13. FCS and Checksum Recalculation	11
14. Behavior of LER/LSRs	12
14.1. Behavior of Timing-capable/aware LERs/LSRs	12
14.2. Behavior of non-Timing-capable/aware LSR	12
15. Other considerations	13
16. Security Considerations	13
17. Applicability Statement	14
18. Acknowledgements	14
19. IANA Considerations	14
20. References	15
20.1. Normative References	15
20.2. Informative References	16
Appendix A. Appendix	17
A.1. Routing extensions for Timing-aware Routers	17
A.2. Signaling Extensions for Creating Timing LSPs	17

Authors' Addresses	18
--------------------	----

1. Introduction

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 RFC2119 [RFC2119].

When used in lower case, these words convey their typical use in common language, and are not to be interpreted as described in RFC2119 [RFC2119].

The objective of timing distribution protocols, such as Precision Time Protocol (PTP) and Network Timing Protocol (NTP), is to synchronize clocks running on nodes of a distributed system.

Timing distribution protocols are presently transported over IP or Ethernet. The present document presents a mechanism for transport over Multiprotocol Label Switched (MPLS) networks. Our solution involves transporting timing messages over dedicated "Timing Label Switched Paths (LSPs)". These are ordinary LSPs that carry timing messages and MAY carry Operations, Administration and Maintenance (OAM) or management messages, but do not carry any other traffic.

Timing LSPs may be established statically or via signaling. When using signaling, extensions to routing protocols (e.g., OSPF, ISIS) are required to enable routers to distribute their timing processing capabilities, and extensions to path set up protocols (e.g., RSVP-TE) are required for establishing the LSPs. All such extensions are beyond the scope of this document.

High accuracy timing distribution requires on-path support, e.g., Transparent Clocks (TCs) or Boundary Clocks (BCs), at intermediate nodes. These intermediate nodes need to recognize and appropriately process timing distribution packets. To facilitate efficient recognition of timing messages transported over MPLS, this document restricts the specific encapsulations to be used.

[IEEE-1588] defines PTP messages for frequency, phase and time synchronization. PTP messages may be transported over UDP/IP (Annex D and E of [IEEE-1588]) or over Ethernet (Annex F of [IEEE-1588]). This document defines two methods to transport PTP messages over MPLS networks.

PTP defines several clock types, including ordinary clocks, boundary clocks, end-to-end transparent clocks, and peer-to-peer transparent clocks. Transparent clocks are situated at intermediate nodes and

update the Correction Field inside PTP messages in order to reflect the time required to transit the node.

[RFC5905] defines NTP messages for clock and time synchronization. NTP messages are transported over UDP/IP. This document defines a method to transport NTP messages over MPLS networks.

It can be expected that only a subset of LSR ports will be capable of processing timing messages. Timing LSPs MUST be set up (either by manual provisioning or via signaling) to traverse these ports. While Timing LSPs are designed to optimize timing distribution, the performance of slave clocks is beyond the scope of this document.

Presently on-path support is only defined for PTP, and therefore much of our discussion will focus on PTP. NTP timing distribution may benefit from transport in a Timing LSP due to prioritization or selection of ports or nodes with minimal delay or delay asymmetry.

2. Terminology

1588: The timing distribution protocol defined in IEEE 1588.

Boundary Clock: A device with one timing port to receive timing messages and at least one port to re-distribute timing messages.

CF: Correction Field, a field inside certain PTP messages that holds the accumulated transit time.

Master Clock: The source of 1588 timing messages to a set of slave clocks.

NTP: The timing distribution protocol defined in RFC 5905.

Ordinary Clock: A master or slave clock. Note that ordinary clocks have only a single PTP port.

PTP: Precision Time Protocol. See 1588.

Slave Clock: A receiver of 1588 timing messages from a master clock.

Timing LSP: An MPLS LSP dedicated to carry timing messages.

Timing messages: Timing distribution protocol messages that are exchanged between clocks.

Timing port: A port on a (master, slave, transparent, or boundary) clock.

Timing PW: A PW within a Timing LSP that is dedicated to carry timing messages.

Transparent Clock: An intermediate node that forwards timing messages while updating their CF.

3. Problem Statement

[IEEE-1588] defines methods for transporting PTP messages over Ethernet and IP networks. [RFC5905] defines a method of transporting NTP messages over IP networks. There is a need to transport timing messages over MPLS networks while supporting the Transparent Clock (TC), Boundary Clock (BC) and Ordinary Clock (OC) functionalities in LER and LSRs of the MPLS network.

There are potentially many ways of transporting timing packets over MPLS. However, it is advisable to limit the number of possible encapsulation options to simplify recognition and processing of timing packets.

The solution herein described transports timing messages over dedicated "Timing Label Switched Paths (LSPs)". Were timing packets to share LSPs with other traffic, intermediate LSRs would be required to perform some deeper inspection to differentiate between timing packets and other packets. The method herein proposed avoids this complexity, and can readily detect all PTP messages (one-step or two-step), and supports ordinary, boundary and transparent clocks.

4. Timing over MPLS Architecture

Timing messages are exchanged between timing ports on ordinary and boundary clocks. Boundary clocks terminate the timing messages and act as master clock for other boundary clocks or slave clocks. End-to-End transparent clocks do not terminate the timing messages but do modify the contents of the timing messages in transit.

OC, BC and TC functionality may be implemented in either LERs or LSRs.

An example is shown in Figure 1, where the LERs act as OCs and are the initiating/terminating points for timing messages. The ingress LER encapsulates timing messages in a Timing LSP and the egress LER terminates this Timing LSP. Intermediate LSRs (only one is shown here) act as TCs, updating the CF of transiting timing messages, as well as performing label switching operations.

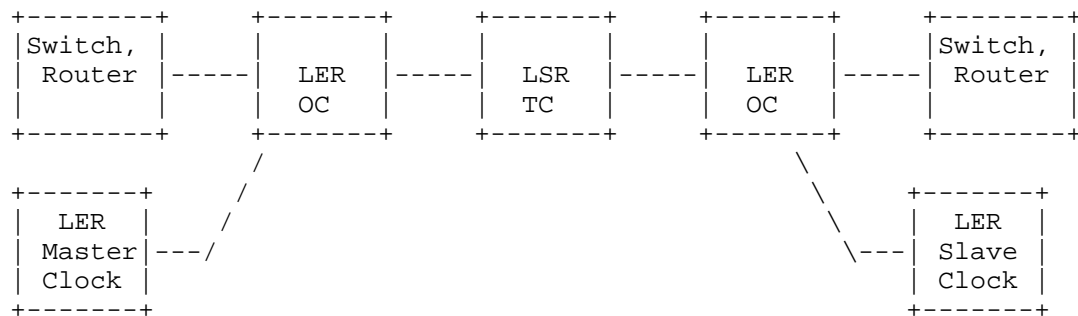


Figure (1) - Deployment example 1 of timing over MPLS network

Another example is shown in Figure 2, where LERs act as BCs, and switches/routers outside of the MPLS network, act as OCs or BCs. The ingress LER BC recovers timing and initiates timing messages encapsulated in the Timing LSP toward the MPLS network, an intermediate LSR acts as a TC, and the egress LER acts as a BC sending timing messages to equipment outside the MPLS network.

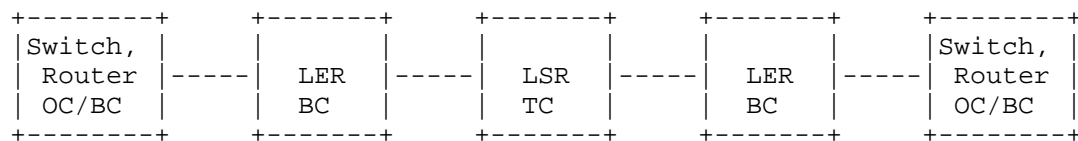


Figure (2) - Deployment example 2 of timing over MPLS network

Yet another example is shown in Figure 3, where both LERs and LSRs act as TCs. The ingress LER updates the CF and encapsulates the timing message in an MPLS packet, intermediate LSRs update the CF and perform label switching, and the egress LER updates the CF and sends the timing messages to equipment outside the MPLS network.

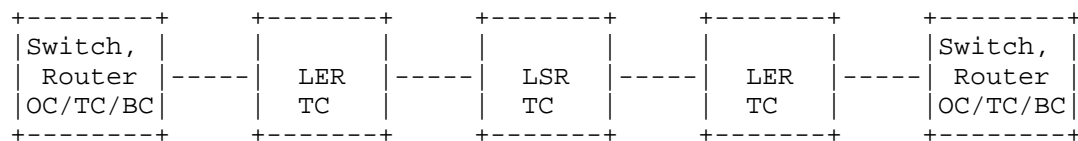


Figure (3) - Deployment example 3 of timing over MPLS network

A final example is shown in Figure 4, where all nodes act as BCs. Single-hop LSPs are created between every two adjacent LSRs. Of course, PTP transport over Ethernet MAY be used between two network elements.

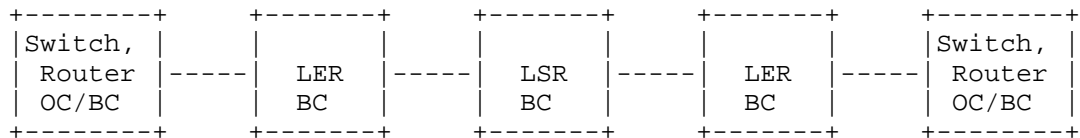


Figure (4) - Deployment example 3 of timing over MPLS network

An MPLS domain MAY serve multiple customers, each having its own Timing domain. In these cases the MPLS domain (maintained by a service provider) MUST provide dedicated timing services to each customer.

The timing over MPLS architecture assumes a full mesh of Timing LSPs between all LERs supporting this specification. It supports point-to-point (VPWS) and Multipoint (VPLS) services. This means that a customer may purchase a point-to-point timing service between two customer sites or a multipoint timing service between more than two customer sites.

The Timing over MPLS architecture supports P2P or P2MP Timing LSPs. This means that the Timing Multicast messages such as PTP Multicast event messages MAY be transported over P2MP Timing LSPs or MAY be replicated and transported over multiple P2P Timing LSPs.

Timing LSPs, as defined by this specification, MAY be used for timing messages that do not require time-stamping or CF updating.

PTP Announce messages that determine the Timing LSP terminating point behavior such as BC/OC/TC SHOULD be transported over the Timing LSP to simplify hardware and software.

5. Dedicated LSPs for Timing messages

The method defined in this document is used by LER and LSRs to identify timing messages by observing the top label of the MPLS label stack. Compliant implementations MUST use dedicated LSPs to carry timing messages over MPLS. Such LSPs are herein referred to as "Timing LSPs" and the labels associated with these LSPs as "Timing LSP labels".

Timing distribution requires symmetrical bidirectional communications. Co-routing of the two directions is required to limit delay asymmetry. Thus timing messages **MUST** be transported either over two co-routed unidirectional Timing LSPs, or a single bidirectional co-routed Timing LSP.

Timing LSPs **MAY** be configured using RSVP-TE. Extensions to RSVP-TE are required for this purpose, but are beyond the scope of this document.

6. Timing over LSP Encapsulation

We define two methods for carrying timing messages over MPLS. The first method transports UDP/IP-encapsulated timing messages over Timing LSPs, and the second method transports Ethernet encapsulated timing messages over Ethernet PWs placed in Timing LSPs.

6.1. Timing over UDP/IP over MPLS Encapsulation

The first method directly encapsulates UDP/IP timing messages in a Timing LSP. The UDP/IP encapsulation of PTP messages **MUST** comply to Annex D and E of [IEEE-1588], and the UDP/IP encapsulation of NTP messages **MUST** comply to [RFC5905]. This format is shown in Figure 4.

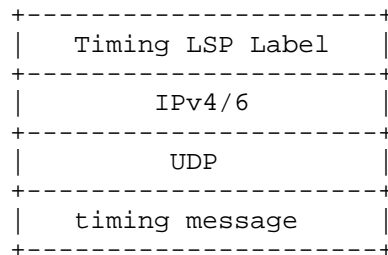


Figure (4) - Timing over UDP/IP over MPLS Encapsulation

In order for an LER/LSR to process timing messages, the Timing LSP Label must be the top label of the label stack. The LER/LSR **MUST** know that this label is a Timing LSP Label. It can learn this by static configuration or via RSVP-TE signaling.

6.2. Timing over PW Encapsulation

Another method of transporting timing over MPLS networks is to use Ethernet encapsulated timing messages, and to transport these in an Ethernet PW which in turn is transported over a Timing LSP. In the

case of PTP, the Ethernet encapsulation MUST comply to Annex F of [IEEE-1588] and the Ethernet PW encapsulation to [RFC4448], resulting in the format shown in Figure 5(A).

Either the Raw mode or Tagged mode defined in [RFC-4448] MAY be used and the payload MAY have 0, 1, or 2 VLAN tags. The Timing over PW encapsulation MUST use the Control Word (CW) as specified in [RFC4448]. The use of Sequence Number in the CW is optional.

NTP MAY be transported using an IP PW (as defined in [RFC4447]) as shown in Fig 5(B).

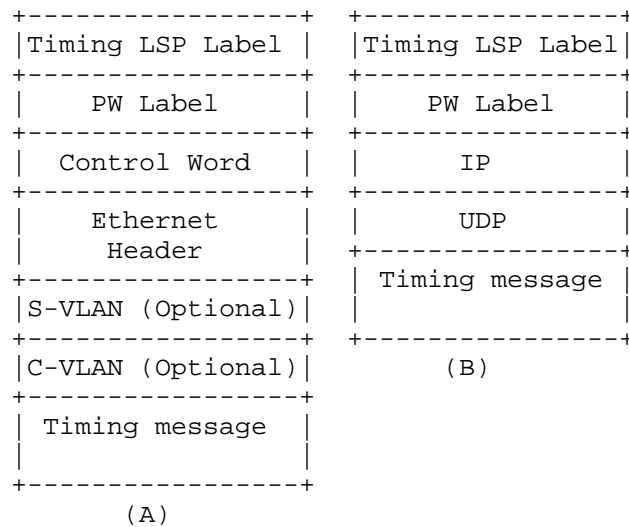


Figure (5) - Timing over PW Encapsulations

7. Timing message Processing

Each Timing protocol such as PTP and NTP, defines a set of timing messages. PTP defines SYNC, DELAY_REQ, DELAY_RESP, FOLLOW_UP, etc.

Some timing messages require per-packet processing, such as time-stamping or CF updating. A compliant LER/LSR parses each timing message to determine the required processing.

For example, the following PTP messages (event messages) require time-stamping or CF updating:

- o SYNC

- o DELAY_REQ (Delay Request)
- o PDELAY_REQ (Peer Delay Request)
- o PDELAY_RESP (Peer Delay Response)

SYNC and DELAY_REQ are exchanged between a Master Clock and a Slave Clock and MUST be transported over Timing LSPs. PDELAY_REQ and PDELAY_RESP are exchanged between adjacent PTP clocks (master, slave, boundary, or transparent) and SHOULD be transported over single hop Timing LSPs. If two-Step PTP clocks are present, then the FOLLOW_UP, and PDELAY_RESP_FOLLOW_UP messages MUST also be transported over Timing LSPs.

For a given instance of the 1588 protocol, SYNC and DELAY_REQ MUST be transported in opposite directions. As aforementioned, two co-routed unidirectional LSPs or a single bidirectional co-routed LSP MAY be used.

Except as indicated above for two-step PTP clocks, PTP messages that are not "event messages" need not be processed by intermediate routers. These message types MAY be carried in PTP Tunnel LSPs.

8. Protection and Redundancy

In order to ensure continuous uninterrupted operation of timing distribution, slave clocks often track redundant master clocks. Prolonged outages of Timing LSPs trigger switching to a redundant master clock. It is the responsibility of the network operator to ensure that physically disjoint Timing LSPs are established between a slave clock and redundant master clocks.

LSP or PW layer protection, such as linear protection Switching, ring protection switching or MPLS Fast Reroute (FRR), will lead to changes in propagation delay between master and slave clocks. Such a change, if undetected by the slave clock, would negatively impact timing performance. While it is expected that slave clocks will often be able to detect such delay changes, this specification RECOMMENDS that automatic protection switching NOT be used for Timing LSPs, unless the operator can ensure that it will not negatively impact timing performance.

9. ECMP and Entropy

To ensure the correct operation of slave clocks and avoid error introduced by forward and reverse path delay asymmetry, the physical path taken by timing messages MUST be the same for all timing

messages. In particular, the PTP event messages listed in section 7 MUST be routed in the same way.

Therefore the Timing LSPs MUST not be subject to ECMP (Equal Cost Multipath). Entropy labels MUST NOT be used for the Timing LSP [RFC6790] and MUST NOT be used for PWs inside the Timing LSP [RFC6391].

10. PHP

To ensure that the label on the top of the label stack is the Timing LSP Label, PHP MUST not be employed.

11. OAM, Control and Management

In order to monitor Timing LSPs or PWs, it is necessary to enable them to carry OAM messages. OAM packets MUST be differentiated from timing messages by already defined IETF methods.

For example BFD [RFC5880], [RFC5884] and LSP-Ping [RFC4389] MAY run over Timing LSPs via UDP/IP encapsulation or via GAL/G-ACh. These protocols can easily be identified by the UDP Destination port number or by GAL/G-ACh respectively.

Also BFD, LSP-Ping and other messages MAY run over Timing PWs via VCCV [RFC5085]. In this case these messages are recognized according to the VCCV type.

12. QoS Considerations

There may be deployments where timing messages traverse LSR/LERs that are not capable of the required processing. In order to minimize the negative impact on the timing performance of the slave clock timing messages MUST be treated with the highest priority. This can be achieved by proper setup of Timing LSPs.

It is recommended that Timing LSPs be configured to indicate EF-PHB [RFC3246] for the CoS and "green" [RFC2697] for drop eligibility.

13. FCS and Checksum Recalculation

Since Boundary and Transparent Clocks modify packets, when the MPLS packets are transported over Ethernet the processing MUST include recalculation of the Ethernet FCS. FCS retention as described in [RFC4720] MUST NOT be used.

For the UDP/IP encapsulation mode, calculation of the UDP checksum will generally be required. After updating the CF a Transparent

Clock MUST either incrementally update the UDP checksum or completely recalculate the checksum before transmission to downstream node.

14. Behavior of LER/LSRs

Timing-aware LERs or LSRs are MPLS routers that are able to recognize timing packets. Timing-capable LERs and LSRs further have one or more interfaces that can perform timing processing (OC/BC/TC) on timing packets. Timing-capable/aware LERs and LSRs MAY advertise the timing capabilities of their interfaces via control plane protocols such as OSPF or IS-IS, and timing-aware LERs can then be set up Timing LSPs via RSVP-TE signaling. Alternatively the timing capabilities of LERs and LSRs may be known by a centralized controller or management system, and Timing LSPs may be manually configured, or set up by a management platform or a Software Defined Networking (SDN) controller.

14.1. Behavior of Timing-capable/aware LERs/LSRs

When a timing-capable ingress LER acting as a TC receives a timing message packet from a timing-capable non-MPLS interface, the LER updates the CF, encapsulates and forwards the packet over a previously established Timing LSP. When a timing-capable egress LER acting as a TC receives a timing message packet on timing-capable MPLS interface, the LER updates the CF, decapsulates the MPLS encapsulation, and forwards the packet via a non-MPLS interface. When a timing-capable LSR acting as a TC receives a timing message from a timing-capable MPLS interface, the LSR updates the CF and forwards the timing message over another MPLS interface.

When a timing-capable LER acting as a BC receives a timing message packet from a timing-capable interface, the LER time-stamps the packet and sends it to the BC processing module.

When a timing-capable LER acting as an OC receives a timing message from a timing-capable MPLS interface, the LER time-stamps the packet and sends it to the OC processing module.

14.2. Behavior of non-Timing-capable/aware LSR

It is most beneficial when all LSRs in the path of a Timing LSP be timing-Capable/aware LSRs. This would ensure the highest quality time and clock synchronization by slave clocks. However, this specification does not mandate that all LSRs in path of a Timing LSP be timing-capable/aware.

Non-timing-capable/aware LSRs just perform label switching on the packets encapsulated in Timing LSPs and don't perform any timing

related processing. However, as explained in QoS section, timing packets MUST be still be treated with the highest priority based on their Traffic Class marking.

15. Other considerations

[IEEE-1588] defines an optional peer-to-peer transparent clocking (P2P TC) mode that compensates both for residence time in the network node and for propagation time on the link between nodes. To support P2P TC, delay measurement must be performed between two adjacent timing-capable/aware LSRs. Thus, in addition to the TC functionality detailed above on transit PTP timing messages, adjacent peer to peer TCs MUST engage in single-hop peer delay measurement.

For single hop peer delay measurement a single-hop LSP SHOULD be created between the two adjacent LSRs. Other methods MAY be used; for example, if the link between the two adjacent routers is Ethernet, PTP transport over Ethernet MAY be used.

To support P2P TC, a timing-capable/ware LSR MUST maintain a list of all neighbors to which it needs to send a PDelay_Req, and maintain a single-hop timing LSP to each.

The use of Explicit Null Label (label 0 or 2) is acceptable as long as either the Explicit Null label is the bottom of stack label (for the UDP/IP encapsulation) or the label below the Explicit Null label (for the PW case).

16. Security Considerations

Security considerations for MPLS and pseudowires are discussed in [RFC3985] and [RFC4447]. Security considerations for timing are discussed in [RFC7384]. Everything discussed in those documents applies to the Timing LSP of this document.

An experimental security protocol is defined in [IEEE-1588]. The PTP security extension and protocol provides group source authentication, message integrity, and replay attack protection for PTP messages.

When the MPLS network (provider network) serves multiple customers, it is important to distinguish between timing messages belonging to different customers. For example if an LER BC is synchronized to a grandmaster belonging to customer A, then the LER MUST only use that BC for slaves of customer A, to ensure that customer A cannot adversely affect the timing distribution of other customers.

Timing messages MAY be encrypted or authenticated, provided that the timing-capable LERs/LSRs can authenticate/ decrypt the timing messages.

17. Applicability Statement

The Timing over MPLS transport methods described in this document apply to the following network Elements:

- o An ingress LER that receives IP or Ethernet encapsulated timing messages from a non-MPLS interface and forwards them as MPLS encapsulated timing messages over Timing LSP, optionally performing TC functionality.
- o An egress LER that receives MPLS encapsulated timing messages from a Timing LSP and forwards them to non-MPLS interface as IP or Ethernet encapsulated timing messages, optionally performing TC functionality.
- o An ingress LER that receives MPLS encapsulated timing messages from a non-MPLS interface, performs BC functionality, and sends timing messages over a Timing LSP.
- o An egress LER that receives MPLS encapsulated timing messages from a Timing LSP, performs BC functionality, and sends timing messages over a non-MPLS interface.
- o An LSR on a Timing LSP that receives MPLS encapsulated timing messages from one MPLS interface and forwards them to another MPLS interface, optionally performing TC functionality.

This document also supports the case where not all LSRs are timing-capable/aware, or not all LER/LSR interfaces are timing-capable/aware.

18. Acknowledgements

The authors would like to thank Yaakov Stein, Luca Martini, Ron Cohen, Tal Mizrahi, Stefano Ruffini, Peter Meyer and other IETF participants for reviewing and providing feedback on this draft.

19. IANA Considerations

There are no IANA requirements in this specification.

20. References

20.1. Normative References

- [IEEE-1588] IEEE 1588-2008, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", July 2008.
- [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>>.
- [RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <<http://www.rfc-editor.org/info/rfc3985>>.
- [RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <<http://www.rfc-editor.org/info/rfc4389>>.
- [RFC4447] Martini, L., Ed., Rosen, E., El-Aawar, N., Smith, T., and G. Heron, "Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)", RFC 4447, DOI 10.17487/RFC4447, April 2006, <<http://www.rfc-editor.org/info/rfc4447>>.
- [RFC4448] Martini, L., Ed., Rosen, E., El-Aawar, N., and G. Heron, "Encapsulation Methods for Transport of Ethernet over MPLS Networks", RFC 4448, DOI 10.17487/RFC4448, April 2006, <<http://www.rfc-editor.org/info/rfc4448>>.
- [RFC4720] Malis, A., Allan, D., and N. Del Regno, "Pseudowire Emulation Edge-to-Edge (PWE3) Frame Check Sequence Retention", RFC 4720, DOI 10.17487/RFC4720, November 2006, <<http://www.rfc-editor.org/info/rfc4720>>.
- [RFC5085] Nadeau, T., Ed. and C. Pignataro, Ed., "Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires", RFC 5085, DOI 10.17487/RFC5085, December 2007, <<http://www.rfc-editor.org/info/rfc5085>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<http://www.rfc-editor.org/info/rfc5880>>.

- [RFC5884] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow, "Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)", RFC 5884, DOI 10.17487/RFC5884, June 2010, <<http://www.rfc-editor.org/info/rfc5884>>.

20.2. Informative References

- [ISO] ISO/IEC 10589:1992, "Intermediate system to Intermediate system routing information exchange protocol for use in conjunction with the Protocol for providing the Connectionless-mode Network Service (ISO 8473)", April 1992.
- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <<http://www.rfc-editor.org/info/rfc1195>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<http://www.rfc-editor.org/info/rfc2328>>.
- [RFC2697] Heinanen, J. and R. Guerin, "A Single Rate Three Color Marker", RFC 2697, DOI 10.17487/RFC2697, September 1999, <<http://www.rfc-editor.org/info/rfc2697>>.
- [RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, DOI 10.17487/RFC3246, March 2002, <<http://www.rfc-editor.org/info/rfc3246>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<http://www.rfc-editor.org/info/rfc5340>>.
- [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>>.
- [RFC6391] Bryant, S., Ed., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, DOI 10.17487/RFC6391, November 2011, <<http://www.rfc-editor.org/info/rfc6391>>.

- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <<http://www.rfc-editor.org/info/rfc6790>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<http://www.rfc-editor.org/info/rfc7384>>.

Appendix A. Appendix

A.1. Routing extensions for Timing-aware Routers

MPLS-TE routing relies on extensions to OSPF [RFC2328] [RFC5340] and IS-IS [ISO] [RFC1195] in order to advertise Traffic Engineering (TE) link information used for constraint-based routing.

Timing related capabilities, such as the capability for a router to perform time-stamping, and OC, TC or BC processing, need to be advertised in order for them to be taken into account during path computation. A management system or SDN controller cognizant of timing related capabilities, can prefer or even require a Timing LSP to traverse links or nodes or interfaces with the required capabilities. The optimal path will optimize the performance of the slave clock.

Extensions are required to OSPF and IS-IS in order to advertise timing related capabilities of a link. Such extensions are outside the scope of this document; however such extensions SHOULD be able to signal the following information per Router Link:

- o Capable of processing PTP, NTP or other timing flows
- o Capable of performing TC operation
- o Capable of performing BC operation

A.2. Signaling Extensions for Creating Timing LSPs

RSVP-TE signaling MAY be used to set up Timing LSPs. Extensions are required to RSVP-TE for this purpose. Such extensions are outside the scope of this document; however, the following information MAY be included in such extensions:

- o Offset from Bottom of Stack (BoS) to the start of the Time-stamp field
- o Number of VLANs in case of PW encapsulation

- o Time-stamp field Type
 - * Correction Field, time-stamp
- o Time-stamp Field format
 - * 64-bit PTPv1, 80-bit PTPv2, 32-bit NTP, 64-bit NTP, 128-bit NTP, etc.

Note that when the above optional information is signaled with RSVP-TE for a Timing LSP, all the timing packets carried in that LSP must have the same signaled characteristics. For example if time-stamp format is signaled as 64-bit PTPv1, then all timing packets must use 64-bit PTPv1 time-stamp.

Authors' Addresses

Shahram Davari
Broadcom Corp.
San Jose, CA 95134
USA

Email: davari@broadcom.com

Amit Oren
Broadcom Corp.
San Jose, CA 95134
USA

Email: amito@broadcom.com

Manav Bhatia
Alcatel-Lucent
Bangalore
India

Email: manav.bhatia@alcatel-lucent.com

Peter Roberts
Alcatel-Lucent
Kanata
Canada

Email: peter.roberts@alcatel-lucent.com

Laurent Montini
Cisco Systems
San Jose CA
USA

Email: lmontini@cisco.com

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-ntp-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/lid-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-ntp-mib

-00 Mar 11 Initial version; showed structure of MIB

draft-ietf-tictoc-ntp-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 "ntpbases"
- corrected some data types
- corrected references and typos

-02 Jul 12 Revised following comment at IETF83:

- changed "ptpbasedClockPortRunningIPversion" to the more generic "ptpbasedClockPortRunningTransport", 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 ptpbasedClockPortCurrentPeerAddressType, ptpbasedClockPortRunningTransport, ptpbasedClockPortAssociateAddressType, 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 ptpbasedClockPortSyncOneStep to ptpbasedClockPortSyncTwoStep 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 DESCRIPTION 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 enumerationUpdated 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 removed 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 "ntp 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 }

ptpbasedClockCurrentDSTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF PtpbasedClockCurrentDSEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "Table of information about the PTP clock Current Datasets for
        all domains."
    ::= { ptpbaseMIBClockInfo 1 }

ptpbasedClockCurrentDSEntry OBJECT-TYPE
    SYNTAX      PtpbasedClockCurrentDSEntry
    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          {
                ptpbasedClockCurrentDSDomainIndex,
                ptpbasedClockCurrentDSClockTypeIndex,
                ptpbasedClockCurrentDSInstanceIndex
            }
    ::= { ptpbasedClockCurrentDSTable 1 }

PtpbasedClockCurrentDSEntry ::= SEQUENCE {
    ptpbasedClockCurrentDSDomainIndex      PtpClockDomainType,
    ptpbasedClockCurrentDSClockTypeIndex    PtpClockType,
    ptpbasedClockCurrentDSInstanceIndex      PtpClockInstanceType,
    ptpbasedClockCurrentDSStepsRemoved       Unsigned32,
    ptpbasedClockCurrentDSOffsetFromMaster   PtpClockTimeInterval,
    ptpbasedClockCurrentDSMeanPathDelay      PtpClockTimeInterval
}

ptpbasedClockCurrentDSDomainIndex 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."
    ::= { ptpbasedClockCurrentDSEntry 1 }

```

ptpbasedClockCurrentDSClockTypeIndex OBJECT-TYPE
SYNTAX PtpClockType
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"This object specifies the clock type as defined in the
Textual convention description."
::= { ptpbasedClockCurrentDSEntry 2 }

ptpbasedClockCurrentDSInstanceIndex 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."
::= { ptpbasedClockCurrentDSEntry 3 }

ptpbasedClockCurrentDSStepsRemoved 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]"
::= { ptpbasedClockCurrentDSEntry 4 }

ptpbasedClockCurrentDSOffsetFromMaster 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]"
::= { ptpbasedClockCurrentDSEntry 5 }

ptpbasedClockCurrentDSMeanPathDelay 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 }

ptpbasedClockParentDSParentStats 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]"

::= { ptpbasedClockParentDSEntry 5 }

ptpbasedClockParentDSOffset 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]"

::= { ptpbasedClockParentDSEntry 6 }

ptpbasedClockParentDSClockPhChRate 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]"

::= { ptpbasedClockParentDSEntry 7 }

ptpbasedClockParentDSGMClockIdentity 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 }

ptpbaseClockParentDSGrandmasterClockQualityOffset 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 }

ptpbasedClockRunningClockTypeIndex OBJECT-TYPE

SYNTAX PtpClockType
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"This object specifies the clock type as defined in the
Textual convention description."

::= { ptpbasedClockRunningEntry 2 }

ptpbasedClockRunningInstanceIndex 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."

::= { ptpbasedClockRunningEntry 3 }

ptpbasedClockRunningState OBJECT-TYPE

SYNTAX PtpClockStateType
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object specifies the Clock state returned by a PTP
engine."

::= { ptpbasedClockRunningEntry 4 }

ptpbasedClockRunningPacketsSent 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."

::= { ptpbasedClockRunningEntry 5 }

ptpbasedClockRunningPacketsReceived 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."

::= { ptpbasedClockRunningEntry 6 }

```

ptpbasedClockTimePropertiesDSTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF PtpbasedClockTimePropertiesDSEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Table of information about the PTP clock time properties
        datasets for all domains."
    ::= { ptpbaseMIBClockInfo 5 }

ptpbasedClockTimePropertiesDSEntry OBJECT-TYPE
    SYNTAX          PtpbasedClockTimePropertiesDSEntry
    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
        {
            ptpbasedClockTimePropertiesDSDomainIndex,
            ptpbasedClockTimePropertiesDSClockTypeIndex,
            ptpbasedClockTimePropertiesDSInstanceIndex
        }
    ::= { ptpbasedClockTimePropertiesDSTable 1 }

PtpbasedClockTimePropertiesDSEntry ::= SEQUENCE {
    ptpbasedClockTimePropertiesDSDomainIndex      PtpClockDomainType,
    ptpbasedClockTimePropertiesDSClockTypeIndex    PtpClockType,
    ptpbasedClockTimePropertiesDSInstanceIndex
PtpClockInstanceType,
    ptpbasedClockTimePropertiesDSCurrentUTCOffsetValid TruthValue,
    ptpbasedClockTimePropertiesDSCurrentUTCOffset      Integer32,
    ptpbasedClockTimePropertiesDSLeap59                 TruthValue,
    ptpbasedClockTimePropertiesDSLeap61                 TruthValue,
    ptpbasedClockTimePropertiesDSTimeTraceable          TruthValue,
    ptpbasedClockTimePropertiesDSFreqTraceable          TruthValue,
    ptpbasedClockTimePropertiesDSPTPTimescale           TruthValue,
    ptpbasedClockTimePropertiesDSSource
PtpClockTimeSourceType
}

ptpbasedClockTimePropertiesDSDomainIndex 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 }
```

ptpbasedClockTimePropertiesDSPTPTimescale 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]"
::= { ptpbasedClockTimePropertiesDSentry 10 }

ptpbasedClockTimePropertiesDSSource 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]"
::= { ptpbasedClockTimePropertiesDSentry 11 }

ptpbasedClockTransDefaultDSTable OBJECT-TYPE

SYNTAX SEQUENCE OF PtpbasedClockTransDefaultDSentry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
 "Table of information about the PTP Transparent clock Default
 Datasets for all domains."
::= { ptpbasedMIBClockInfo 6 }

ptpbasedClockTransDefaultDSentry OBJECT-TYPE

SYNTAX PtpbasedClockTransDefaultDSentry
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
 {
 ptpbasedClockTransDefaultDSDomainIndex,
 ptpbasedClockTransDefaultDSInstanceIndex
 }

```
 ::= { ptptimeClockTransDefaultDSTable 1 }

PtpptimeClockTransDefaultDSEntry ::= SEQUENCE {
    ptptimeClockTransDefaultDSDomainIndex  PtpClockDomainType,
    ptptimeClockTransDefaultDSInstanceIndex PtpClockInstanceType,
    ptptimeClockTransDefaultDSClockIdentity PtpClockIdentity,
    ptptimeClockTransDefaultDSNumOfPorts    Counter32,
    ptptimeClockTransDefaultDSDelay         PtpClockMechanismType,
    ptptimeClockTransDefaultDSPrimaryDomain PtpClockDomainType
}

ptptimeClockTransDefaultDSDomainIndex 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."
    ::= { ptptimeClockTransDefaultDSEntry 1 }

ptptimeClockTransDefaultDSInstanceIndex 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."
    ::= { ptptimeClockTransDefaultDSEntry 2 }

ptptimeClockTransDefaultDSClockIdentity 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]"
    ::= { ptptimeClockTransDefaultDSEntry 3 }

ptptimeClockTransDefaultDSNumOfPorts 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 }

```

ptpbasedClockPortTablePortNumberIndex OBJECT-TYPE

SYNTAX PtpClockPortNumber

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This object specifies the PTP Portnumber for this port."

::= { ptpbasedClockPortEntry 4 }

ptpbasedClockPortName 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."

::= { ptpbasedClockPortEntry 5 }

ptpbasedClockPortRole OBJECT-TYPE

SYNTAX PtpClockRoleType

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object describes the current role (slave/master) of the port."

::= { ptpbasedClockPortEntry 6 }

ptpbasedClockPortSyncTwoStep 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."

::= { ptpbasedClockPortEntry 7 }

ptpbasedClockPortCurrentPeerAddressType 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."

::= { ptpbasedClockPortEntry 8 }

ptpbasedClockPortCurrentPeerAddress 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 }
```

ptpbasedClockPortRunningTable OBJECT-TYPE

SYNTAX SEQUENCE OF PtpbasedClockPortRunningEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION
 "Table of information about the clock ports running datasets for
 a particular domain."
 ::= { ptpbasedMIBClockInfo 9 }

ptpbasedClockPortRunningEntry OBJECT-TYPE

SYNTAX PtpbasedClockPortRunningEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION
 "An entry in the table, containing running dataset information
 about a single clock port."
 INDEX {
 ptpbasedClockPortRunningDomainIndex,
 ptpbasedClockPortRunningClockTypeIndex,
 ptpbasedClockPortRunningClockInstanceIndex,
 ptpbasedClockPortRunningPortNumberIndex
 }
 ::= { ptpbasedClockPortRunningTable 1 }

PtpbasedClockPortRunningEntry ::= SEQUENCE {
 ptpbasedClockPortRunningDomainIndex PtpClockDomainType,
 ptpbasedClockPortRunningClockTypeIndex PtpClockType,
 ptpbasedClockPortRunningClockInstanceIndex PtpClockInstanceType,
 ptpbasedClockPortRunningPortNumberIndex PtpClockPortNumber,
 ptpbasedClockPortRunningName DisplayString,
 ptpbasedClockPortRunningState PtpClockPortState,
 ptpbasedClockPortRunningRole PtpClockRoleType,
 ptpbasedClockPortRunningInterfaceIndex InterfaceIndexOrZero,
 ptpbasedClockPortRunningTransport AutonomousType,
 ptpbasedClockPortRunningEncapsulationType AutonomousType,
 ptpbasedClockPortRunningTxMode PtpClockTxModeType,
 ptpbasedClockPortRunningRxMode PtpClockTxModeType,
 ptpbasedClockPortRunningPacketsReceived Counter64,
 ptpbasedClockPortRunningPacketsSent Counter64
 }

ptpbasedClockPortRunningDomainIndex 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 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 }

ptpbaseTransportTypeIPv4 OBJECT-IDENTITY

STATUS current

DESCRIPTION

"IP version 4"

::= { ptpbaseWellKnownTransportTypes 1 }

ptpbaseTransportTypeIPv6 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 }
```

ntpClockPortCurrentClockTypeIndex OBJECT-TYPE

SYNTAX PtpClockType
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"This object specifies the given port's clock type."
::= { ptpbaseClockPortAssociateEntry 2 }

ntpClockPortCurrentClockInstanceIndex 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 }

ntpClockPortCurrentPortNumberIndex 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 }
```

ptpbasesMIBClockPortDSGroup 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 }
```

ptpbasesMIBClockPortRunningGroup 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 }
```

ptpbasesMIBClockPortTransDSGroup 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,
```

ptpbasedClockCurrentDSOffsetFromMaster,
ptpbasedClockCurrentDSMeanPathDelay expose the values of a clock's
current dataset (currentDS).

ptpbasedClockParentDSParentPortIdentity,
ptpbasedClockParentDSParentStats,
ptpbasedClockParentDSOffset,
ptpbasedClockParentDSClockPhChRate,
ptpbasedClockParentDSGMClockIdentity,
ptpbasedClockParentDSGMClockPriority1,
ptpbasedClockParentDSGMClockPriority2,
ptpbasedClockParentDSGMClockQualityClass,
ptpbasedClockParentDSGMClockQualityAccuracy,
ptpbasedClockParentDSGMClockQualityOffset expose the values of a
clock's parent dataset (parentDS).

ptpbasedClockDefaultDSTwoStepFlag,
ptpbasedClockDefaultDSClockIdentity,
ptpbasedClockDefaultDSPriority1,
ptpbasedClockDefaultDSPriority2,
ptpbasedClockDefaultDSSlaveOnly,
ptpbasedClockDefaultDSQualityClass,
ptpbasedClockDefaultDSQualityAccuracy,
ptpbasedClockDefaultDSQualityOffset expose the values of a clock's
default dataset (defaultDS).

ptpbasedClockTimePropertiesDSCurrentUTCOffsetValid,
ptpbasedClockTimePropertiesDSCurrentUTCOffset,
ptpbasedClockTimePropertiesDSLeap59,
ptpbasedClockTimePropertiesDSLeap61,
ptpbasedClockTimePropertiesDSTimeTraceable,
ptpbasedClockTimePropertiesDSFreqTraceable,
ptpbasedClockTimePropertiesDSPTPTimescale,
ptpbasedClockTimePropertiesDSSource expose the values of a clock's
time properties dataset (timePropertiesDS).

ptpbasedClockTransDefaultDSClockIdentity,
ptpbasedClockTransDefaultDSNumOfPorts,
ptpbasedClockTransDefaultDSDelay,
ptpbasedClockTransDefaultDSPrimaryDomain expose the values of a
transparent clock's default dataset (transparentClockDefaultDS).

ptpbasedClockPortName,
ptpbasedClockPortRole,
ptpbasedClockPortSyncTwoStep,
ptpbasedClockPortCurrentPeerAddressType,
ptpbasedClockPortCurrentPeerAddress,
ptpbasedClockPortNumOfAssociatedPorts expose general information
about a clock port.

ptpbasedClockPortRunningName,
ptpbasedClockPortRunningState,
ptpbasedClockPortRunningRole,
ptpbasedClockPortRunningInterfaceIndex,
ptpbasedClockPortRunningTransport,
ptpbasedClockPortRunningEncapsulationType,
ptpbasedClockPortRunningTxMode,
ptpbasedClockPortRunningRxMode,
ptpbasedClockPortRunningPacketsReceived,
ptpbasedClockPortRunningPacketsSent expose a clock port's current running status.

ptpbasedClockPortDSName,
ptpbasedClockPortDSPortIdentity,
ptpbasedClockPortDSlogAnnouncementInterval,
ptpbasedClockPortDSAnnounceRctTimeout,
ptpbasedClockPortDSlogSyncInterval,
ptpbasedClockPortDSMinDelayReqInterval,
ptpbasedClockPortDSPeerDelayReqInterval,
ptpbasedClockPortDSDelayMech,
ptpbasedClockPortDSPeerMeanPathDelay,
ptpbasedClockPortDSGrantDuration,
ptpbasedClockPortDSPTPVersion expose the values of a clock port's port dataset (portDS).

ptpbasedClockPortTransDSPortIdentity,
ptpbasedClockPortTransDSlogMinPdelayReqInt,
ptpbasedClockPortTransDSFaultyFlag,
ptpbasedClockPortTransDSPeerMeanPathDelay expose the values of a transparent clock port's port dataset (transparentClockPortDS).

ptpbasedClockPortAssociateAddressType,
ptpbasedClockPortAssociateAddress,
ptpbasedClockPortAssociatePacketsSent,
ptpbasedClockPortAssociatePacketsReceived,
ptpbasedClockPortAssociateInErrors,
ptpbasedClockPortAssociateOutErrors 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
-----	-----
ptpbasesMIB	{ mib-2 xxx }

[NOTE for IANA: Please allocate an object identifier at <http://www.iana.org/assignments/smi-numbers> for object ptpbasesMIB.]

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

TICTOC Working Group
Internet Draft
Intended status: Informational
Expires: March 2015

T. Mizrahi
Marvell

September 3, 2014

Security Requirements of Time Protocols
in Packet Switched Networks
draft-ietf-tictoc-security-requirements-12.txt

Abstract

As time and frequency distribution protocols are becoming increasingly common and widely deployed, concern about their exposure to various security threats is increasing. This document defines a set of security requirements for time protocols, focusing on the Precision Time Protocol (PTP) and the Network Time Protocol (NTP). This document also discusses the security impacts of time protocol practices, the performance implications of external security practices on time protocols and the dependencies between other security services and time synchronization.

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/lid-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 3, 2015.

Copyright Notice

Copyright (c) 2014 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	3
2. Conventions Used in this Document	5
2.1. Terminology	5
2.2. Abbreviations	5
2.3. Common Terminology for PTP and NTP	6
2.4. Terms used in this Document	6
3. Security Threats	7
3.1. Threat Model	7
3.1.1. Internal vs. External Attackers	7
3.1.2. Man in the Middle (MITM) vs. Packet Injector	8
3.2. Threat Analysis.....	8
3.2.1. Packet Manipulation	8
3.2.2. Spoofing	9
3.2.3. Replay Attack	9
3.2.4. Rogue Master Attack	9
3.2.5. Packet Interception and Removal	10
3.2.6. Packet Delay Manipulation	10
3.2.7. L2/L3 DoS Attacks	10
3.2.8. Cryptographic Performance Attacks	10
3.2.9. DoS Attacks against the Time Protocol	10
3.2.10. Grandmaster Time Source Attack (e.g., GPS fraud) ..	11
3.2.11. Exploiting Vulnerabilities in the Time Protocol ..	11
3.2.12. Network Reconnaissance	11
3.3. Threat Analysis Summary	11
4. Requirement Levels	13
5. Security Requirements	14
5.1. Clock Identity Authentication and Authorization	14
5.1.1. Authentication and Authorization of Masters	15
5.1.2. Recursive Authentication and Authorization of Masters (Chain of Trust)	16

5.1.3. Authentication and Authorization of Slaves	17
5.1.4. PTP: Authentication and Authorization of P2P TCs by the Master	17
5.1.5. PTP: Authentication and Authorization of Control Messages	18
5.2. Protocol Packet Integrity	19
5.2.1. PTP: Hop-by-hop vs. End-to-end Integrity Protection	20
5.2.1.1. Hop-by-Hop Integrity Protection	20
5.2.1.2. End-to-End Integrity Protection	20
5.3. Spoofing Prevention	21
5.4. Availability	22
5.5. Replay Protection	22
5.6. Cryptographic Keys and Security Associations	23
5.6.1. Key Freshness	23
5.6.2. Security Association	23
5.6.3. Unicast and Multicast Associations	24
5.7. Performance	25
5.8. Confidentiality.....	26
5.9. Protection against Packet Delay and Interception Attacks	26
5.10. Combining Secured with Unsecured Nodes	27
5.10.1. Secure Mode	27
5.10.2. Hybrid Mode	28
6. Summary of Requirements	29
7. Additional security implications	30
7.1. Security and on-the-fly Timestamping	31
7.2. PTP: Security and Two-Step Timestamping	31
7.3. Intermediate Clocks	31
7.4. External Security Protocols and Time Protocols.....	32
7.5. External Security Services Requiring Time	33
7.5.1. Timestamped Certificates	33
7.5.2. Time Changes and Replay Attacks	33
8. Issues for Further Discussion	33
9. Security Considerations	34
10. IANA Considerations.....	34
11. Acknowledgments	34
12. References	34
12.1. Normative References	34
12.2. Informative References	34
13. Contributing Authors	36

1. Introduction

As time protocols are becoming increasingly common and widely deployed, concern about the resulting exposure to various security threats is increasing. If a time protocol is compromised, the applications it serves are prone to a range of possible attacks including Denial-of-Service (DoS) or incorrect behavior.

This document discusses the security aspects of time distribution protocols in packet networks, and focuses on the two most common protocols, the Network Time Protocol [NTPv4] and the Precision Time Protocol (PTP) [IEEE1588]. Note, that although PTP was not defined by the IETF, it is one of the two most common time protocols and hence it is included in the discussion.

The Network Time Protocol was defined with an inherent security protocol; [NTPv4] defines a security protocol that is based on a symmetric key authentication scheme, and [AutoKey] presents an alternative security protocol, based on a public key authentication scheme. [IEEE1588] includes an experimental security protocol, defined in Annex K of the standard, but this Annex was never formalized into a fully defined security protocol.

While NTP includes an inherent security protocol, the absence of a standard security solution for PTP undoubtedly contributed to the wide deployment of unsecured time synchronization solutions. However, in some cases security mechanisms may not be strictly necessary, e.g., due to other security practices in place, or due to the architecture of the network. A time synchronization security solution, much like any security solution, is comprised of various building blocks, and must be carefully tailored for the specific system it is deployed in. Based on a system-specific threat assessment, the benefits of a security solution must be weighed against the potential risks, and based on this tradeoff an optimal security solution can be selected.

The target audience of this document includes:

- o Timing and networking equipment vendors - can benefit from this document by deriving the security features that should be supported in the time/networking equipment.
- o Standard development organizations - can use the requirements defined in this document when specifying security mechanisms for a time protocol.
- o Network operators - can use this document as a reference when designing the network and its security architecture. As stated above, the requirements in this document may be deployed selectively based on a careful per-system threat analysis.

This document attempts to add clarity to the time protocol security requirements discussion by addressing a series of questions:

(1) What are the threats that need to be addressed for the time protocol, and thus what security services need to be provided? (e.g. a malicious NTP server or PTP master)

(2) What external security practices impact the security and performance of time keeping, and what can be done to mitigate these impacts? (e.g. an IPsec tunnel in the time protocol traffic path)

(3) What are the security impacts of time protocol practices? (e.g. on-the-fly modification of timestamps)

(4) What are the dependencies between other security services and time protocols? (e.g. which comes first - the certificate or the timestamp?)

In light of the questions above, this document defines a set of requirements for security solutions for time protocols, focusing on PTP and NTP.

2. Conventions Used in this Document

2.1. Terminology

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 [KEYWORDS].

This document describes security requirements, and thus requirements are phrased in the document in the form "the security mechanism MUST/SHOULD/...". Note, that the phrasing does not imply that this document defines a specific security mechanism, but defines the requirements with which every security mechanism should comply.

2.2. Abbreviations

BC	Boundary Clock [IEEE1588]
DoS	Denial of Service
MITM	Man In The Middle
NTP	Network Time Protocol [NTPv4]
OC	Ordinary Clock [IEEE1588]
P2P TC	Peer-to-Peer Transparent Clock [IEEE1588]

PTP Precision Time Protocol [IEEE1588]

TC Transparent Clock [IEEE1588]

2.3. Common Terminology for PTP and NTP

This document refers to both PTP and NTP. For the sake of consistency, throughout the document the term "master" applies to both a PTP master and an NTP server. Similarly, the term "slave" applies to both PTP slaves and NTP clients. The term "protocol packets" refers generically to PTP and NTP messages.

2.4. Terms used in this Document

- o Clock - A node participating in the protocol (either PTP or NTP). A clock can be a master, a slave, or an intermediate clock (see corresponding definitions below).
- o Control packets - Packets used by the protocol to exchange information between clocks that is not strictly related to the time. NTP uses NTP Control Messages. PTP uses Announce, Signaling and Management messages.
- o End-to-end security - A security approach where secured packets sent from a source to a destination are not modified by intermediate nodes, allowing the destination to authenticate the source of the packets, and to verify their integrity. In the context of confidentiality, end-to-end encryption guarantees that intermediate nodes cannot eavesdrop to en-route packets. However, as discussed in Section 5. , confidentiality is not a strict requirement in this document.
- o Grandmaster - A master that receives time information from a locally attached clock device, and not through the network. A grandmaster distributes its time to other clocks in the network.
- o Hop-by-hop security - A security approach where secured packets sent from a source to a destination may be modified by intermediate nodes. In this approach intermediate nodes share the encryption key with the source and destination, allowing them to re-encrypt or re-authenticate modified packets before relaying them to the destination.
- o Intermediate clock - A clock that receives timing information from a master, and sends timing information to other clocks. In NTP this term refers to an NTP server that is not a Stratum 1 server. In PTP this term refers to a BC or a TC.

- o Master - A clock that generates timing information to other clocks in the network.
In NTP 'master' refers to an NTP server. In PTP 'master' refers to a master OC (aka grandmaster) or to a port of a BC that is in the master state.
- o Protocol packets - Packets used by the time protocol. The terminology used in this document distinguishes between time packets and control packets.
- o Secured clock - A clock that supports a security mechanism that complies to the requirements in this document.
- o Slave - A clock that receives timing information from a master. In NTP 'slave' refers to an NTP client. In PTP 'slave' refers to a slave OC, or to a port of a BC that is in the slave state.
- o Time packets - Protocol packets carrying time information.
- o Unsecured clock - A clock that does not support a security mechanism according to the requirements in this document.

3. Security Threats

This section discusses the possible attacker types and analyzes various attacks against time protocols.

The literature is rich with security threats of time protocols, e.g., [Traps], [AutoKey], [TimeSec], [SecPTP], and [SecSen]. The threat analysis in this document is mostly based on [TimeSec].

3.1. Threat Model

A time protocol can be attacked by various types of attackers.

The analysis in this document classifies attackers according to 2 criteria, as described in Section 3.1.1. and Section 3.1.2.

3.1.1. Internal vs. External Attackers

In the context of internal and external attackers, the underlying assumption is that the time protocol is secured either by an encryption or an authentication mechanism, or both.

Internal attackers either have access to a trusted segment of the network, or possess the encryption or authentication keys. An internal attack can also be performed by exploiting vulnerabilities

in devices; for example, by installing malware, or obtaining credentials to reconfigure the device. Thus, an internal attacker can maliciously tamper with legitimate traffic in the network, as well as generate its own traffic and make it appear legitimate to its attacked nodes.

Note that internal attacks are a special case of Byzantine failures, where a node in the system may fail in arbitrary ways; by crashing, by omitting messages, or by malicious behavior. This document focuses on nodes that demonstrate malicious behavior.

External attackers, on the other hand, do not have the keys, and have access only to the encrypted or authenticated traffic.

Obviously, in the absence of a security mechanism there is no distinction between internal and external attackers, since all attackers are internal in practice.

3.1.2. Man in the Middle (MITM) vs. Packet Injector

MITM attackers are located in a position that allows interception and modification of in-flight protocol packets. It is assumed that an MITM attacker has physical access to a segment of the network, or has gained control of one of the nodes in the network.

A traffic injector is not located in an MITM position, but can attack by generating protocol packets. An injector can reside either within the attacked network, or on an external network that is connected to the attacked network. An injector can also potentially eavesdrop on protocol packets sent as multicast, record them and replay them later.

3.2. Threat Analysis

3.2.1. Packet Manipulation

A packet manipulation attack results when an MITM attacker receives timing protocol packets, alters them and relays them to their destination, allowing the attacker to maliciously tamper with the protocol. This can result in a situation where the time protocol is apparently operational but providing intentionally inaccurate information.

3.2.2. Spoofing

In spoofing, an injector masquerades as a legitimate node in the network by generating and transmitting protocol packets or control packets. Two typical examples of spoofing attacks:

- o An attacker can impersonate the master, allowing malicious distribution of false timing information.
- o An attacker can impersonate a legitimate clock, a slave or an intermediate clock, by sending malicious messages to the master, causing the master to respond to the legitimate clock with protocol packets that are based on the spoofed messages. Consequently, the delay computations of the legitimate clock are based on false information.

As with packet manipulation, this attack can result in a situation where the time protocol is apparently operational but providing intentionally inaccurate information.

3.2.3. Replay Attack

In a replay attack, an attacker records protocol packets and replays them at a later time without any modification. This can also result in a situation where the time protocol is apparently operational but providing intentionally inaccurate information.

3.2.4. Rogue Master Attack

In a rogue master attack, an attacker causes other nodes in the network to believe it is a legitimate master. As opposed to the spoofing attack, in the Rogue Master attack the attacker does not fake its identity, but rather manipulates the master election process using malicious control packets. For example, in PTP, an attacker can manipulate the Best Master Clock Algorithm (BMCA), and cause other nodes in the network to believe it is the most eligible candidate to be a grandmaster.

In PTP, a possible variant of this attack is the rogue TC/BC attack. Similar to the rogue master attack, an attacker can cause victims to believe it is a legitimate TC or BC, allowing the attacker to manipulate the time information forwarded to the victims.

3.2.5. Packet Interception and Removal

A packet interception and removal attack results when an MITM attacker intercepts and drops protocol packets, preventing the destination node from receiving some or all of the protocol packets.

3.2.6. Packet Delay Manipulation

In a packet delay manipulation scenario, an MITM attacker receives protocol packets, and relays them to their destination after adding a maliciously computed delay. The attacker can use various delay attack strategies; the added delay can be constant, jittered, or slowly wandering. Each of these strategies has a different impact, but they all effectively manipulate the attacked clock.

Note that the victim still receives one copy of each packet, contrary to the replay attack, where some or all of the packets may be received by the victim more than once.

3.2.7. L2/L3 DoS Attacks

There are many possible Layer 2 and Layer 3 DoS attacks, e.g., IP spoofing, ARP spoofing [Hack], MAC flooding [Anatomy], and many others. As the target's availability is compromised, the timing protocol is affected accordingly.

3.2.8. Cryptographic Performance Attacks

In cryptographic performance attacks, an attacker transmits fake protocol packets, causing high utilization of the cryptographic engine at the receiver, which attempts to verify the integrity of these fake packets.

This DoS attack is applicable to all encryption and authentication protocols. However, when the time protocol uses a dedicated security mechanism implemented in a dedicated cryptographic engine, this attack can be applied to cause DoS specifically to the time protocol.

3.2.9. DoS Attacks against the Time Protocol

An attacker can attack a clock by sending an excessive number of time protocol packets, thus degrading the victim's performance. This attack can be implemented, for example, using the attacks described in Section 3.2.2. and Section 3.2.4.

3.2.10. Grandmaster Time Source Attack (e.g., GPS fraud)

Grandmasters receive their time from an external accurate time source, such as an atomic clock or a GPS clock, and then distribute this time to the slaves using the time protocol.

Time source attack are aimed at the accurate time source of the grandmaster. For example, if the grandmaster uses a GPS based clock as its reference source, an attacker can jam the reception of the GPS signal, or transmit a signal similar to one from a GPS satellite, causing the grandmaster to use a false reference time.

Note that this attack is outside the scope of the time protocol. While various security measures can be taken to mitigate this attack, these measures are outside the scope of the security requirements defined in this document.

3.2.11. Exploiting Vulnerabilities in the Time Protocol

Time protocols can be attacked by exploiting vulnerabilities in the protocol, implementation bugs, or misconfigurations (e.g., [NTPDDoS]). It should be noted that such attacks cannot typically be mitigated by security mechanisms. However, when a new vulnerability is discovered, operators should react as soon as possible, and take the necessary measures to address it.

3.2.12. Network Reconnaissance

An attacker can exploit the time protocol to collect information such as addresses and locations of nodes that take part in the protocol. Reconnaissance can be applied either by passively eavesdropping to protocol packets, or by sending malicious packets and gathering information from the responses. By eavesdropping to a time protocol, an attacker can learn the network latencies, which provide information about the network topology and node locations.

Moreover, properties such as the frequency of the protocol packets, or the exact times at which they are sent can allow fingerprinting of specific nodes; thus, protocol packets from a node can be identified even if network addresses are hidden or encrypted.

3.3. Threat Analysis Summary

The two key factors to a threat analysis are the impact and the likelihood of each of the analyzed attacks.

Table 1 summarizes the security attacks presented in Section 3.2. For each attack, the table specifies its impact, and its applicability to each of the attacker types presented in Section 3.1.

Table 1 clearly shows the distinction between external and internal attackers, and motivates the usage of authentication and integrity protection, significantly reducing the impact of external attackers.

The Impact column provides an intuitive measure of the severity of each attack, and the relevant Attacker Type columns provide an intuition about how difficult each attack is to implement, and hence about the likelihood of each attack.

The impact column in Table 1 can have one of 3 values:

- o DoS - the attack causes denial of service to the attacked node, the impact of which is not restricted to the time protocol.
- o Accuracy degradation - the attack yields a degradation in the slave accuracy, but does not completely compromise the slaves' time and frequency.
- o False time - slaves align to a false time or frequency value due to the attack. Note that if the time protocol aligns to a false time, it may cause DoS to other applications that rely on accurate time. However, for the purpose of the analysis in this section we distinguish this implication from 'DoS', which refers to a DoS attack that is not necessarily aimed at the time protocol. All attacks that have a '+' for 'False Time' implicitly have a '+' for 'Accuracy Degradation'. Note, that 'False Time' necessarily implies 'Accuracy Degradation'. However, two different terms are used, indicating two levels of severity.

The Attacker Type columns refer to the 4 possible combinations of the attacker types defined in Section 3.1.

Attack	Impact			Attacker Type			
	False Time	Accuracy Degrad.	DoS	Internal MITM	External MITM	Internal Inj.	External Inj.
Manipulation	+			+			

Spoofing	+		+	+		
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Replay attack	+		+	+		
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Rogue master attack	+		+	+		
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Interception and removal		+	+	+	+	
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Packet delay manipulation	+		+		+	
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
L2/L3 DoS attacks			+	+	+	+
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Crypt. performance attacks			+	+	+	+
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Time protocol DoS attacks			+	+	+	
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Master time source attack (e.g., GPS spoofing)	+			+	+	+
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

Table 1 Threat Analysis - Summary

The threats discussed in this section provide the background for the security requirements presented in Section 5.

4. Requirement Levels

The security requirements are presented in Section 5. Each requirement is defined with a requirement level, in accordance with the requirement levels defined in Section 2.1.

The requirement levels in this document are affected by the following factors:

- o Impact:

The possible impact of not implementing the requirement, as illustrated in the 'impact' column of Table 1.
For example, a requirement that addresses a threat that can be implemented by an external injector is typically a 'MUST', since the threat can be implemented by all the attacker types analyzed in Section 3.1.

- o Difficulty of the corresponding attack:
The level of difficulty of the possible attacks that become possible by not implementing the requirement. The level of difficulty is reflected in the 'Attacker Type' column of Table 1. For example, a requirement that addresses a threat that only compromises the availability of the protocol is typically no more than a 'SHOULD'.
- o Practical considerations:
Various practical factors that may affect the requirement. For example, if a requirement is very difficult to implement, or is applicable to very specific scenarios, these factors may reduce the requirement level.

Section 5. lists the requirements. For each requirement there is a short explanation detailing the reason for its requirement level.

5. Security Requirements

This section defines a set of security requirements. These requirements are phrased in the form "the security mechanism MUST/SHOULD/MAY...". However, this document does not specify how these requirements can be met. While these requirements can be satisfied by defining explicit security mechanisms for time protocols, at least a subset of the requirements can be met by applying common security practices to the network or by using existing security protocols, such as [IPsec] or [MACsec]. Thus, security solutions that address these requirements are outside the scope of this document.

5.1. Clock Identity Authentication and Authorization

Requirement

The security mechanism MUST support authentication.

Requirement

The security mechanism MUST support authorization.

Requirement Level

The requirements in this subsection address the spoofing attack (Section 3.2.2.), and the rogue master attack (Section 3.2.4.).

The requirement level of these requirements is 'MUST' since in the absence of these requirements the protocol is exposed to attacks that are easy to implement and have a high impact.

Discussion

Authentication refers to verifying the identity of the peer clock. Authorization, on the other hand, refers to verifying that the peer clock is permitted to play the role that it plays in the protocol. For example, some nodes may be permitted to be masters, while other nodes are only permitted to be slaves or TCs.

Authentication is typically implemented by means of a cryptographic signature, allowing to verify the identity of the sender. Authorization requires clocks to maintain a list of authorized clocks, or a "black list" of clocks that should be denied service or revoked.

It is noted that while the security mechanism is required to provide an authorization mechanism, the deployment of such a mechanism depends on the nature of the network. For example, a network that deploys PTP may consist of a set of identical OCs, where all clocks are equally permitted to be a master. In such a network an authorization mechanism may not be necessary.

The following subsections describe 5 distinct cases of clock authentication.

5.1.1. Authentication and Authorization of Masters

Requirement

The security mechanism MUST support an authentication mechanism, allowing slaves to authenticate the identity of masters.

Requirement

The authentication mechanism MUST allow slaves to verify that the authenticated master is authorized to be a master.

Requirement Level

The requirements in this subsection address the spoofing attack (Section 3.2.2.), and the rogue master attack (Section 3.2.4.).

The requirement level of these requirements is 'MUST' since in the absence of these requirements the protocol is exposed to attacks that are easy to implement and have a high impact.

Discussion

Clocks authenticate masters in order to ensure the authenticity of the time source. It is important for a slave to verify the identity of the master, as well as to verify that the master is indeed authorized to be a master.

5.1.2. Recursive Authentication and Authorization of Masters (Chain of Trust)

Requirement

The security mechanism MUST support recursive authentication and authorization of the master, to be used in cases where time information is conveyed through intermediate clocks.

Requirement Level

The requirement in this subsection addresses the spoofing attack (Section 3.2.2.), and the rogue master attack (Section 3.2.4.).

The requirement level of this requirement is 'MUST' since in the absence of this requirement the protocol is exposed to attacks that are easy to implement and have a high impact.

Discussion

In some cases a slave is connected to an intermediate clock, that is not the primary time source. For example, in PTP a slave can be connected to a Boundary Clock (BC) or a Transparent Clock (TC), which in turn is connected to a grandmaster. A similar example in NTP is when a client is connected to a stratum 2 server, which is connected to a stratum 1 server. In both the PTP and the NTP cases, the slave authenticates the intermediate clock, and the intermediate clock authenticates the grandmaster. This recursive authentication process is referred to in [AutoKey] as proventionation.

Specifically in PTP, this requirement implies that if a slave receives time information through a TC, it must authenticate the TC it is attached to, as well as authenticate the master it receives the time information from, as per Section 5.1.1. Similarly, if a TC receives time information through an attached TC, it must authenticate the attached TC.

5.1.3. Authentication and Authorization of Slaves

Requirement

The security mechanism MAY provide a means for a master to authenticate its slaves.

Requirement

The security mechanism MAY provide a means for a master to verify that the sender of a protocol packet is authorized to send a packet of this type.

Requirement Level

The requirement in this subsection prevents DoS attacks against the master (Section 3.2.9.).

The requirement level of this requirement is 'MAY' since:

- o Its low impact, i.e., in the absence of this requirement the protocol is only exposed to DoS.
- o Practical considerations: requiring an NTP server to authenticate its clients may significantly impose on the server's performance.

Note that while the requirement level of this requirement is 'MAY', the requirement in Section 5.1.1. is 'MUST'; the security mechanism must provide a means for authentication and authorization, with an emphasis on the master. Authentication and authorization of slaves is specified in this subsection as 'MAY'.

Discussion

Slaves and intermediate clocks are authenticated by masters in order to verify that they are authorized to receive timing services from the master.

Authentication of slaves prevents unauthorized clocks from receiving time services. Preventing the master from serving unauthorized clocks can help in mitigating DoS attacks against the master. Note that the authentication of slaves might put a higher load on the master than serving the unauthorized clock, and hence this requirement is a MAY.

5.1.4. PTP: Authentication and Authorization of P2P TCs by the Master

Requirement

The security mechanism for PTP MAY provide a means for a master to authenticate the identity of the P2P TCs directly connected to it.

Requirement

The security mechanism for PTP MAY provide a means for a master to verify that P2P TCs directly connected to it are authorized to be TCs.

Requirement Level

The requirement in this subsection prevents DoS attacks against the master (Section 3.2.9.).

The requirement level of this requirement is 'MAY' for the same reasons specified in Section 5.1.3.

Discussion

P2P TCs that are one hop from the master use the PDelay_Req and PDelay_Resp handshake to compute the link delay between the master and TC. These TCs are authenticated by the master.

Authentication of TCs, much like authentication of slaves, reduces unnecessary load on the master and peer TCs, by preventing the master from serving unauthorized clocks.

5.1.5. PTP: Authentication and Authorization of Control Messages

Requirement

The security mechanism for PTP MUST support authentication of Announce messages. The authentication mechanism MUST also verify that the sender is authorized to be a master.

Requirement

The security mechanism for PTP MUST support authentication and authorization of Management messages.

Requirement

The security mechanism MAY support authentication and authorization of Signaling messages.

Requirement Level

The requirements in this subsection address the spoofing attack (Section 3.2.2.), and the rogue master attack (Section 3.2.4.).

The requirement level of the first two requirements is 'MUST' since in the absence of these requirements the protocol is exposed to attacks that are easy to implement and have a high impact.

The requirement level of the third requirement is 'MAY' since its impact greatly depends on the application for which the Signaling messages are used for.

Discussion

Master election is performed in PTP using the Best Master Clock Algorithm (BMCA). Each Ordinary Clock (OC) announces its clock attributes using Announce messages, and the best master is elected based on the information gathered from all the candidates. Announce messages must be authenticated in order to prevent rogue master attacks (Section 3.2.4.). Note, that this subsection specifies a requirement that is not necessarily included in Section 5.1.1. or in Section 5.1.3. , since the BMCA is initiated before clocks have been defined as masters or slaves.

Management messages are used to monitor or configure PTP clocks. Malicious usage of Management messages enables various attacks, such as the rogue master attack, or DoS attack.

Signaling messages are used by PTP clocks to exchange information that is not strictly related to time information or to master selection, such as unicast negotiation. Authentication and authorization of Signaling message may be required in some systems, depending on the application these messages are used for.

5.2. Protocol Packet Integrity

Requirement

The security mechanism MUST protect the integrity of protocol packets.

Requirement Level

The requirement in this subsection addresses the packet manipulation attack (Section 3.2.1.).

The requirement level of this requirement is 'MUST' since in the absence of this requirement the protocol is exposed to attacks that are easy to implement and have high impact.

Discussion

While Section 5.1. refers to ensuring the identity and authorization of the source of a protocol packet, this subsection refers to ensuring that the packet arrived intact. The integrity protection mechanism ensures the authenticity and completeness of data from the data originator.

Integrity protection is typically implemented by means of an Integrity Check Value (ICV) that is included in protocol packets and is verified by the receiver.

5.2.1. PTP: Hop-by-hop vs. End-to-end Integrity Protection

Specifically in PTP, when protocol packets are subject to modification by TCs, the integrity protection can be enforced in one of two approaches, end-to-end or hop-by-hop.

5.2.1.1. Hop-by-Hop Integrity Protection

Each hop that needs to modify a protocol packet:

- o Verifies its integrity.
- o Modifies the packet, i.e., modifies the correctionField.
Note: Transparent Clocks (TCs) improve the end-to-end accuracy by updating a "correctionField" (clause 6.5 in [IEEE1588]) in the PTP packet by adding the latency caused by the current TC.
- o Re-generates the integrity protection, e.g., re-computes a Message Authentication Code.

In the hop-by-hop approach, the integrity of protocol packets is protected by induction on the path from the originator to the receiver.

This approach is simple, but allows rogue TCs to modify protocol packets.

5.2.1.2. End-to-End Integrity Protection

In this approach, the integrity protection is maintained on the path from the originator of a protocol packet to the receiver. This allows

the receiver to directly validate the protocol packet without the ability of intermediate TCs to manipulate the packet.

Since TCs need to modify the correctionField, a separate integrity protection mechanism is used specifically for the correctionField.

The end-to-end approach limits the TC's impact to the correctionField alone, while the rest of the protocol packet is protected on an end-to-end basis. It should be noted that this approach is more difficult to implement than the hop-by-hop approach, as it requires the correctionField to be protected separately from the other fields of the packet, possibly using different cryptographic mechanisms and keys.

5.3. Spoofing Prevention

Requirement

The security mechanism MUST provide a means to prevent master spoofing.

Requirement

The security mechanism MUST provide a means to prevent slave spoofing.

Requirement

PTP: The security mechanism MUST provide a means to prevent P2P TC spoofing.

Requirement Level

The requirements in this subsection address spoofing attacks (Section 3.2.2.). As described in Section 3.2.2. , when these requirements are not met, the attack may have a high impact, causing slaves to rely on false time information. Thus, the requirement level is 'MUST'.

Discussion

Spoofing attacks may take various different forms, and can potentially cause significant impact. In a master spoofing attack, the attacker causes slaves to receive false information about the current time by masquerading as the master.

By spoofing a slave or an intermediate node (the second example of Section 3.2.2.), an attacker can tamper with the slaves' delay computations. These attacks can be mitigated by an authentication mechanism (Section 5.1.3. and 5.1.4.), or by other means, for example, a PTP Delay_Req can include a Message Authentication Code (MAC) that is included in the corresponding Delay_Resp message, allowing the slave to verify that the Delay_Resp was not sent in response to a spoofed message.

5.4. Availability

Requirement

The security mechanism SHOULD include measures to mitigate DoS attacks against the time protocol.

Requirement Level

The requirement in this subsection prevents DoS attacks against the protocol (Section 3.2.9.).

The requirement level of this requirement is 'SHOULD' due to its low impact, i.e., in the absence of this requirement the protocol is only exposed to DoS.

Discussion

The protocol availability can be compromised by several different attacks. An attacker can inject protocol packets to implement the spoofing attack (Section 3.2.2.) or the rogue master attack (Section 3.2.4.), causing DoS to the victim (Section 3.2.9.).

An authentication mechanism (Section 5.1.) limits these attacks strictly to internal attackers, and thus prevents external attackers from performing them. Hence, the requirements of Section 5.1. can be used to mitigate this attack. Note, that Section 5.1. addresses a wider range of threats, whereas the current section is focused on availability.

The DoS attacks described in Section 3.2.7. are performed at lower layers than the time protocol layer, and are thus outside the scope of the security requirements defined in this document.

5.5. Replay Protection

Requirement

The security mechanism **MUST** include a replay prevention mechanism.

Requirement Level

The requirement in this subsection prevents replay attacks (Section 3.2.3.).

The requirement level of this requirement is 'MUST' since in the absence of this requirement the protocol is exposed to attacks that are easy to implement and have a high impact.

Discussion

The replay attack (Section 3.2.3.) can compromise both the integrity and availability of the protocol. Common encryption and authentication mechanisms include replay prevention mechanisms that typically use a monotonously increasing packet sequence number.

5.6. Cryptographic Keys and Security Associations

5.6.1. Key Freshness

Requirement

The security mechanism **MUST** provide a means to refresh the cryptographic keys.

The cryptographic keys **MUST** be refreshed frequently.

Requirement Level

The requirement level of this requirement is 'MUST' since key freshness is an essential property for cryptographic algorithms, as discussed below.

Discussion

Key freshness guarantees that both sides share a common updated secret key. It also helps in preventing replay attacks. Thus, it is important for keys to be refreshed frequently. Note that the term 'frequently' is used without a quantitative requirement, as the precise frequency requirement should be considered on a per-system basis, based on the threats and system requirements.

5.6.2. Security Association

Requirement

The security protocol SHOULD support a security association protocol where:

- o Two or more clocks authenticate each other.
- o The clocks generate and agree on a cryptographic session key.

Requirement

Each instance of the association protocol SHOULD produce a different session key.

Requirement Level

The requirement level of this requirement is 'SHOULD' since it may be expensive in terms of performance, especially in low-cost clocks.

Discussion

The security requirements in Section 5.1. and Section 5.2. require usage of cryptographic mechanisms, deploying cryptographic keys. A security association (e.g., [IPsec]) is an important building block in these mechanisms.

It should be noted that in some cases different security association mechanisms may be used at different levels of clock hierarchies. For example, the association between a Stratum 2 clock and a Stratum 3 clock in NTP may have different characteristics than an association between two clocks at the same stratum level. On a related note, in some cases a hybrid solution may be used, where a subset of the network is not secured at all (see Section 5.10.2.).

5.6.3. Unicast and Multicast Associations

Requirement

The security mechanism SHOULD support security association protocols for unicast and for multicast associations.

Requirement Level

The requirement level of this requirement is 'SHOULD' since it may be expensive in terms of performance, especially for low-cost clocks.

Discussion

A unicast protocol requires an association protocol between two clocks, whereas a multicast protocol requires an association protocol among two or more clocks, where one of the clocks is a master.

5.7. Performance

Requirement

The security mechanism **MUST** be designed in such a way that it does not significantly degrade the quality of the time transfer.

Requirement

The mechanism **SHOULD** minimize computational load.

Requirement

The mechanism **SHOULD** minimize storage requirements of client state in the master.

Requirement

The mechanism **SHOULD** minimize the bandwidth overhead required by the security protocol.

Requirement Level

While the quality of the time transfer is clearly a 'MUST', the other 3 performance requirements are 'SHOULD', since some systems may be more sensitive to resource consumption than others, and hence these requirements should be considered on a per-system basis.

Discussion

Performance efficiency is important since client restrictions often dictate a low processing and memory footprint, and because the server may have extensive fan-out.

Note that the performance requirements refer to a time-protocol-specific security mechanism. In systems where a security protocol is used for other types of traffic as well, this document does not place any performance requirements on the security protocol performance. For example, if IPsec encryption is used for securing all information between the master and slave node, including information that is not part of the time protocol, the requirements in this subsection are not necessarily applicable.

5.8. Confidentiality

Requirement

The security mechanism MAY provide confidentiality protection of the protocol packets.

Requirement Level

The requirement level of this requirement is 'MAY' since the absence of this requirement does not expose the protocol to severe threats, as discussed below.

Discussion

In the context of time protocols, confidentiality is typically of low importance, since timing information is typically not considered secret information.

Confidentiality can play an important role when service providers charge their customers for time synchronization services, and thus an encryption mechanism can prevent eavesdroppers from obtaining the service without payment. Note that these cases are, for now, rather esoteric.

Confidentiality can also prevent an MITM attacker from identifying protocol packets. Thus, confidentiality can assist in protecting the timing protocol against MITM attacks such as packet delay (Section 3.2.6.), manipulation and interception and removal attacks. Note, that time protocols have predictable behavior even after encryption, such as packet transmission rates and packet lengths. Additional measures can be taken to mitigate encrypted traffic analysis by random padding of encrypted packets and by adding random dummy packets. Nevertheless, encryption does not prevent such MITM attacks, but rather makes these attacks more difficult to implement.

5.9. Protection against Packet Delay and Interception Attacks

Requirement

The security mechanism MUST include means to protect the protocol from MITM attacks that degrade the clock accuracy.

Requirement Level

The requirements in this subsection address MITM attacks such as the packet delay attack (Section 3.2.6.) and packet interception attacks (Section 3.2.5. and Section 3.2.1.).

The requirement level of this requirement is 'MUST'. In the absence of this requirement the protocol is exposed to attacks that are easy to implement and have a high impact. Note that in the absence of this requirement, the impact is similar to packet manipulation attacks (Section 3.2.1.), and thus this requirement has the same requirement level as integrity protection (Section 5.2.).

It is noted that the implementation of this requirement depends on the topology and properties of the system.

Discussion

While this document does not define specific security solutions, we note that common practices for protection against MITM attacks use redundant masters (e.g. [NTPv4]), or redundant paths between the master and slave (e.g. [DelayAtt]). If one of the time sources indicates a time value that is significantly different than the other sources, it is assumed to be erroneous or under attack, and is therefore ignored.

Thus, MITM attack prevention derives a requirement from the security mechanism, and a requirement from the network topology. While the security mechanism should support the ability to detect delay attacks, it is noted that in some networks it is not possible to provide the redundancy needed for such a detection mechanism.

5.10. Combining Secured with Unsecured Nodes

Integrating a security mechanism into a time synchronized system is a complex and expensive process, and hence in some cases may require incremental deployment, where new equipment supports the security mechanism, and is required to interoperate with legacy equipment without the security features.

5.10.1. Secure Mode

Requirement

The security mechanism MUST support a secure mode, where only secured clocks are permitted to take part in the time protocol. In this mode every protocol packet received from an unsecured clock MUST be discarded.

Requirement Level

The requirement level of this requirement is 'MUST' since the full capacity of the security requirements defined in this document can only be achieved in secure mode.

Discussion

While the requirement in this subsection is similar to the one in 5.1. , it refers to the secure mode, as opposed to the hybrid mode presented in the next subsection.

5.10.2. Hybrid Mode

Requirement

The security protocol SHOULD support a hybrid mode, where both secured and unsecured clocks are permitted to take part in the protocol.

Requirement Level

The requirement level of this requirement is a 'SHOULD'; on one hand hybrid mode enables a gradual transition from unsecured to secured mode, which is especially important in large-scaled deployments. On the other hand, hybrid mode is not required in all systems; this document recommends to deploy the 'Secure Mode' described in Section 5.10.1. where possible.

Discussion

The hybrid mode allows both secured and unsecured clocks to take part in the time protocol. NTP, for example, allows a mixture of secured and unsecured nodes.

Requirement

A master in the hybrid mode SHOULD be a secured clock.

A secured slave in the hybrid mode SHOULD discard all protocol packets received from unsecured clocks.

Requirement Level

The requirement level of this requirement is a 'SHOULD', since it may not be applicable to all deployments. For example, a hybrid network may require the usage of unsecured masters or TCs.

Discussion

This requirement ensures that the existence of unsecured clocks does not compromise the security provided to secured clocks. Hence, secured slaves only "trust" protocol packets received from a secured clock.

An unsecured slave can receive protocol packets either from unsecured clocks, or from secured clocks. Note that the latter does not apply when encryption is used. When integrity protection is used, the unsecured slave can receive secured packets ignoring the integrity protection.

Note that the security scheme in [NTPv4] with [AutoKey] does not satisfy this requirement, since nodes prefer the server with the most accurate clock, which is not necessarily the server that supports authentication. For example, a stratum 2 server is connected to two stratum 1 servers, Server A, supporting authentication, and server B, without authentication. If server B has a more accurate clock than A, the stratum 2 server chooses server B, in spite of the fact it does not support authentication.

6. Summary of Requirements

Section	Requirement	Type
5.1.	Authentication & authorization of sender.	MUST
	Authentication & authorization of master.	MUST
	Recursive authentication & authorization.	MUST
	Authentication & authorization of slaves.	MAY
	PTP: Authentication & authorization of P2P TCs by master.	MAY
	PTP: Authentication & authorization of Announce messages.	MUST
	PTP: Authentication & authorization of Management messages.	MUST

	PTP: Authentication & authorization of Signaling messages.	MAY
5.2.	Integrity protection.	MUST
5.3.	Spoofing prevention.	MUST
5.4.	Protection from DoS attacks against the time protocol.	SHOULD
5.5.	Replay protection.	MUST
5.6.	Key freshness.	MUST
	Security association.	SHOULD
	Unicast and multicast associations.	SHOULD
5.7.	Performance: no degradation in quality of time transfer.	MUST
	Performance: computation load.	SHOULD
	Performance: storage.	SHOULD
	Performance: bandwidth.	SHOULD
5.8.	Confidentiality protection.	MAY
5.9.	Protection against delay and interception attacks.	MUST
5.10.	Secure mode.	MUST
	Hybrid mode.	SHOULD

Table 2 Summary of Security Requirements

7. Additional security implications

This section discusses additional implications of the interaction between time protocols and security mechanisms.

This section refers to time protocol security mechanisms, as well as to "external" security mechanisms, i.e., security mechanisms that are not strictly related to the time protocol.

7.1. Security and on-the-fly Timestamping

Time protocols often require that protocol packets be modified during transmission. Both NTP and PTP in one-step mode require clocks to modify protocol packets based on the time of transmission and/or reception.

In the presence of a security mechanism, whether encryption or integrity protection:

- o During transmission the encryption and/or integrity protection MUST be applied after integrating the timestamp into the packet.

To allow high accuracy, timestamping is typically performed as close to the transmission or reception time as possible. However, since the security engine must be placed between the timestamping function and the physical interface, it may introduce non-deterministic latency that causes accuracy degradation. These performance aspects have been analyzed in literature, e.g., [1588IPsec] and [Tunnell].

7.2. PTP: Security and Two-Step Timestamping

PTP supports a two-step mode of operation, where the time of transmission of protocol packets is communicated without modifying the packets. As opposed to one-step mode, two-step timestamping can be performed without the requirement to encrypt after timestamping.

Note that if an encryption mechanism such as IPsec is used, it presents a challenge to the timestamping mechanism, since time protocol packets are encrypted when traversing the physical interface, and are thus impossible to identify. A possible solution to this problem [IPsecSync] is to include an indication in the encryption header that identifies time protocol packets.

7.3. Intermediate Clocks

A time protocol allows slaves to receive time information from an accurate time source. Time information is sent over a path that often traverses one or more intermediate clocks.

- o In NTP, time information originated from a stratum 1 server can be distributed to stratum 2 servers, and in turn distributed from the stratum 2 servers to NTP clients. In this case, the stratum 2 servers are a layer of intermediate clocks. These intermediate clocks are referred to as "secondary servers" in [NTPv4].
- o In PTP, BCs and TCs are intermediate nodes used to improve the accuracy of time information conveyed between the grandmaster and the slaves.

A common rule of thumb in network security is that end-to-end security is the best policy, as it secures the entire path between the data originator and its receiver. The usage of intermediate nodes implies that if a security mechanism is deployed in the network, a hop-by-hop security scheme must be used, since intermediate nodes must be able to send time information to the slaves, or to modify time information sent through them.

This inherent property of using intermediate clocks increases the system's exposure to internal threats, as there is a large number of nodes that possess the security keys.

Thus, there is a tradeoff between the achievable clock accuracy of a system, and the robustness of its security solution. On one hand high clock accuracy calls for hop-by-hop involvement in the protocol, also known as on-path support. On the other hand, a robust security solution calls for end-to-end data protection.

7.4. External Security Protocols and Time Protocols

Time protocols are often deployed in systems that use security mechanisms and protocols.

A typical example is the 3GPP Femtocell network [3GPP], where IPsec is used for securing traffic between a Femtocell and the Femto Gateway. In some cases, all traffic between these two nodes may be secured by IPsec, including the time protocol traffic. This use-case is thoroughly discussed in [IPsecSync].

Another typical example is the usage of MACsec encryption ([MACsec]) in L2 networks that deploy time synchronization [AvbAssum].

The usage of external security mechanisms may affect time protocols as follows:

- o Timestamping accuracy can be affected, as described in 7.1.

- o If traffic is secured between two nodes in the network, no intermediate clocks can be used between these two nodes. In the [3GPP] example, if traffic between the Femtocell and the Femto Gateway is encrypted, then time protocol packets are necessarily transported over the underlying network without modification, and thus cannot enjoy the improved accuracy provided by intermediate clock nodes.

7.5. External Security Services Requiring Time

Cryptographic protocols often use time as an important factor in the cryptographic algorithm. If a time protocol is compromised, it may consequently expose the security protocols that rely on it to various attacks. Two examples are presented in this section.

7.5.1. Timestamped Certificates

Certificate validation requires the sender and receiver to be roughly time synchronized. Thus, synchronization is required for establishing security protocols such as IKEv2 and TLS. Other authentication and key exchange mechanisms, such as Kerberos, also require the parties involved to be synchronized [Kerb].

An even stronger interdependence between a time protocol and a security mechanism is defined in [AutoKey], which defines mutual dependence between the acquired time information, and the authentication protocol that secures it. This bootstrapping behavior results from the fact that trusting the received time information requires a valid certificate, and validating a certificate requires knowledge of the time.

7.5.2. Time Changes and Replay Attacks

A successful attack on a time protocol may cause the attacked clocks to go back in time. The erroneous time may expose cryptographic algorithms that rely on time, as a node may use a key that was already used in the past and has expired.

8. Issues for Further Discussion

The Key distribution is outside the scope of this document. Although this is an essential element of any security system, it is outside the scope of this document.

9. Security Considerations

The security considerations of network timing protocols are presented throughout this document.

10. IANA Considerations

There are no new IANA considerations implied by this document.

11. Acknowledgments

The authors gratefully acknowledge Stefano Ruffini, Doug Arnold, Kevin Gross, Dieter Sibold, Dan Grossman, Laurent Montini, Russell Smiley, Shawn Emery, Dan Romascanu, Stephen Farrell, Kathleen Moriarty, and Joel Jaeggli for their thorough review and helpful comments. The authors would also like to thank members of the TICTOC WG for providing feedback on the TICTOC mailing list.

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

12. References

12.1. Normative References

- [IEEE1588] IEEE TC 9 Instrumentation and Measurement Society, "1588 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Version 2", IEEE Standard, 2008.
- [KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [NTPv4] Mills, D., Martin, J., Burbank, J., Kasch, W., "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

12.2. Informative References

- [1588IPsec] A. Treytl, B. Hirschler, "Securing IEEE 1588 by IPsec tunnels - An analysis", in Proceedings of 2010 International Symposium for Precision Clock Synchronization for Measurement, Control and Communication, ISPCS 2010, pp. 83-90, 2010.
- [3GPP] 3GPP, "Security of Home Node B (HNB) / Home evolved Node B (HeNB)", 3GPP TS 33.320 10.4.0 (work in progress), 2011.

- [Anatomy] C. Nachreiner, "Anatomy of an ARP Poisoning Attack", 2003.
- [AutoKey] Haberman, B., Mills, D., "Network Time Protocol Version 4: Autokey Specification", RFC 5906, June 2010.
- [AvbAssum] D. Pannell, "Audio Video Bridging Gen 2 Assumptions", IEEE 802.1 AVB Plenary, work in progress, May 2012.
- [DelayAtt] T. Mizrahi, "A Game Theoretic Analysis of Delay Attacks against Time Synchronization Protocols", accepted, to appear in Proceedings of the International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication, ISPCS, 2012.
- [Hack] S. McClure, J. Scambray, G. Kurtz, Kurtz, "Hacking exposed: network security secrets and solutions", McGraw-Hill, 2009.
- [IPsec] S. Kent, K. Seo, "Security Architecture for the Internet Protocol", IETF, RFC 4301, 2005.
- [IPsecSync] Y. Xu, "IPsec security for packet based synchronization", IETF, draft-xu-tictoc-ipsec-security-for-synchronization (work in progress), 2011.
- [Kerb] S. Sakane, K. Kamada, M. Thomas, J. Vilhuber, "Kerberosized Internet Negotiation of Keys (KINK)", 2006.
- [MACsec] IEEE 802.1AE-2006, "IEEE Standard for Local and Metropolitan Area Networks - Media Access Control (MAC) Security", 2006.
- [NTPDDoS] "Attackers use NTP reflection in huge DDoS attack", TICTOC mail archive, 2014.
- [SecPTP] J. Tsang, K. Beznosov, "A security analysis of the precise time protocol (short paper)," 8th International Conference on Information and Communication Security (ICICS 2006), pp. 50-59, 2006.

- [SecSen] S. Ganeriwal, C. Popper, S. Capkun, M. B. Srivastava, "Secure Time Synchronization in Sensor Networks", ACM Trans. Info. and Sys. Sec., Volume 11, Issue 4, July 2008.
- [TimeSec] T. Mizrahi, "Time synchronization security using IPsec and MACsec", ISPCS 2011, pp. 38-43, 2011.
- [Traps] Treytl, A., Gaderer, G., Hirschler, B., Cohen, R., "Traps and pitfalls in secure clock synchronization" in Proceedings of 2007 International Symposium for Precision Clock Synchronization for Measurement, Control and Communication, ISPCS 2007, pp. 18-24, 2007.
- [Tunnel] A. Treytl, B. Hirschler, and T. Sauter, "Secure tunneling of high precision clock synchronisation protocols and other timestamped data", in Proceedings of the 8th IEEE International Workshop on Factory Communication Systems (WFCS), vol. ISBN 978-1-4244-5461-7, pp. 303-313, 2010.

13. Contributing Authors

Karen O'Donoghue
ISOC

Email: odonoghue@isoc.org

Authors' Addresses

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam, 20692 Israel

Email: talmi@marvell.com

NTP Working Group
Internet Draft
Intended status: Standards Track
Updates: 5905
Expires: July 2014

T. Mizrahi
Marvell
D. Mayer
Network Time Foundation
January 2, 2014

Using NTP Extension Fields without Authentication
draft-mizrahi-ntp-extension-field-03.txt

Abstract

The Network Time Protocol Version 4 (NTPv4) defines the optional usage of extension fields. An extension field is an optional field that resides at the end of the NTP header, and can be used to add optional capabilities or additional information that is not conveyed in the standard NTP header. The current definition of extension fields in NTPv4 is somewhat ambiguous regarding the connection between extension fields and the presence of a Message Authentication Code (MAC). This draft clarifies the usage of extension fields in the presence and in the absence of a MAC, while maintaining interoperability with existing implementations.

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/lid-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 2, 2014.

Copyright Notice

Copyright (c) 2014 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	3
2. Conventions Used in this Document	4
2.1. Terminology	4
2.2. Terms & Abbreviations	4
3. NTP Extension Fields with and without a MAC - Clarifications .	4
3.1. Extension Field Format	4
3.2. Extension Fields in the Absence of a MAC	4
3.3. Unknown Extension Fields	5
3.4. Interoperability with Current Implementations	5
4. NTP Extension Field Usage with and without a MAC - Extensions	5
4.1. Extension Fields in the Presence of a MAC	5
4.2. Extension Fields in the Absence of a MAC	5
4.3. Multiple Extension fields in an NTP packet	6
4.4. MAC in the absence of an Extension field	6
5. Security Considerations	6
6. IANA Considerations	6
7. Acknowledgments	6
8. References	6
8.1. Normative References	6
8.2. Informative References	7
Appendix A. Requirements from NTPv4 and Autokey	7
A.1. NTP Extension Field for Future Extensions	7
A.2. NTP Extension Field in the Presence of a MAC	7
A.3. The NTP Extension Field Format	7
A.4. NTP Extension Field in Autokey	8

1. Introduction

The NTP header format consists of a set of fixed fields that may be followed by some optional fields. Two types of optional fields are defined, Message Authentication Codes (MAC), and extension fields.

If a MAC is used, it resides at the end of the packet. This field can be either 24 octets long, 20 octets long, or a 4-octet crypto-NAK.

NTP extension fields were defined in [RFC5905] as a generic mechanism that allows to add future extensions and features without modifying the NTP header format.

The only currently defined extension field is the one used by the AutoKey protocol [RFC5906].

The NTP specification is somewhat ambiguous with regards to the connection between using extension fields and the presence of a MAC.

- o The definition of the NTP extension field implies that it was intended to be a generic mechanism that can be used for various future features of the protocol (see Section A.1.).
- o On the other hand, the NTP extension field description in [RFC5905] states that a MAC is always present when an extension field is present (see Section A.2.).

The last two quotes seem to be in contradiction; since the extension field was defined as a generic future-compatible building block, it seems unlikely to bind it to a specific feature in the protocol.

Moreover, the extension field parsing rules presented in [RFC5906] imply that an extension field can be present without a MAC, provided that the extension field is at least 28 Octets long.

This document attempts to resolve the ambiguity with regards to the connection between NTP extension fields and MACs, updating Section 7.5 of [RFC5905], and describes the usage of extension fields in the absence of a MAC in a way that is interoperable with current implementations.

2. Conventions Used in this Document

2.1. Terminology

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 [KEYWORDS].

2.2. Terms & Abbreviations

NTPv4 Network Time Protocol Version 4

MAC Message Authentication Code

3. NTP Extension Fields with and without a MAC - Clarifications

This section clarifies the usage of extension fields in the absence of a MAC, in accordance with the definitions in [RFC5905] and [RFC5906]. Section 4. defines a more generic and flexible usage of extension fields.

3.1. Extension Field Format

The NTP extension field is defined in Section 7.5 of [RFC5905]. The extension field format is quoted here in Section A.3.

The minimal length of an extension field, as defined in Section 7.5 of [RFC5905], is 16 octets.

3.2. Extension Fields in the Absence of a MAC

Extension fields can be used when a MAC is not present in the NTP packet. In this case, the extension fields must comply with the parsing rules in Section A.4. Specifically:

- o If the packet includes a single extension field, the length of the extension field MUST be at least 7 words, i.e., at least 28 octets.
- o If the packet includes more than one extension field, the length of the last extension field MUST be at least 28 octets. The length of the other extension fields in this case MUST be at least 16 octets each, as defined in [RFC5905].

A host that supports NTP extension fields MUST parse NTP extension fields as described in Section A.4.

3.3. Unknown Extension Fields

If an extension field is unknown to the receiving server the server should ignore the extension field and may optionally drop the packet altogether if policy requires it. Note that in the presence of an unknown extension field any MAC that may be present may be misinterpreted as an unknown extension though in this case the apparent extension length will be totally inconsistent with the total length of the rest of the packet.

3.4. Interoperability with Current Implementations

The behavior described in Section 3.2. is compliant to [RFC5906], and thus should be compatible with existing implementations that support NTP extension fields.

4. NTP Extension Field Usage with and without a MAC - Extensions

This section updates [RFC5905] and [RFC5906] with respect to the usage of extension fields, allowing a more flexible and unambiguous usage.

4.1. Extension Fields in the Presence of a MAC

The usage of extension fields in the presence of a MAC is specified in [RFC5905] and in [RFC5906]. The requirement for a MAC MUST be specified by the specification for the extension field and the specification MUST include both the algorithm to be used to create the MAC and the length of the MAC thus created. An extension field may allow for more than one algorithm to be used in which case the information about which one was used MUST be included in the extension field itself.

4.2. Extension Fields in the Absence of a MAC

Extension fields can be used when a MAC is not present in the NTP packet. In this case, the extension fields must comply with the following:

- o If the packet includes a single extension field, the length of the extension field MUST be at least 16 octets. The extension length is specified in the length field of the extension and is the number of octets in the extension field.

- o If the packet includes more than one extension field, the length of the last extension field MUST be at least 28 octets. The length of the other extension fields in this case MUST be at least 16 octets each, as defined in [RFC5905].

4.3. Multiple Extension fields in an NTP packet

If there are multiple extension fields that require a MAC they MUST all require use of the same algorithm and MAC length. Extension fields that do not require a MAC can be included with extension fields that do require a MAC.

4.4. MAC in the absence of an Extension field

A MAC must not be any longer than 24 octets if there is no extension field present unless through a previous exchange of packets with an extension field which defines the size and algorithm of the MAC transmitted in the packet and is agreed upon by both client and server.

5. Security Considerations

The security considerations of the network time protocol are discussed in [RFC5905]. This document clarifies some ambiguity with regards to the usage of the NTP extension field, and thus the behavior described in this document does not introduce new security considerations.

6. IANA Considerations

There are no new IANA considerations implied by this document.

7. Acknowledgments

The authors thank Dave Mills for his insightful comments.

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

8. References

8.1. Normative References

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

- [RFC5905] Mills, D., Martin, J., Burbank, J., Kasch, W.,
"Network Time Protocol Version 4: Protocol and
Algorithms Specification", RFC 5905, June 2010.

8.2. Informative References

- [RFC5906] Haberman, B., Mills, D., "Network Time Protocol
Version 4: Autokey Specification", RFC 5906, June
2010.

Appendix A.

Requirements from NTPv4 and Autokey

A.1. NTP Extension Field for Future Extensions

The following paragraph is quoted from Section 16 of [RFC5905].

This document introduces NTP extension fields allowing for the development of future extensions to the protocol, where a particular extension is to be identified by the Field Type sub-field within the extension field.

A.2. NTP Extension Field in the Presence of a MAC

The following paragraph is quoted from Section 7.5 of [RFC5905].

In NTPv4, one or more extension fields can be inserted after the header and before the MAC, which is always present when an extension field is present.

A.3. The NTP Extension Field Format

Figure 1 specifies the NTP extension field format, and is quoted from [RFC5905]. For further details refer to [RFC5905].

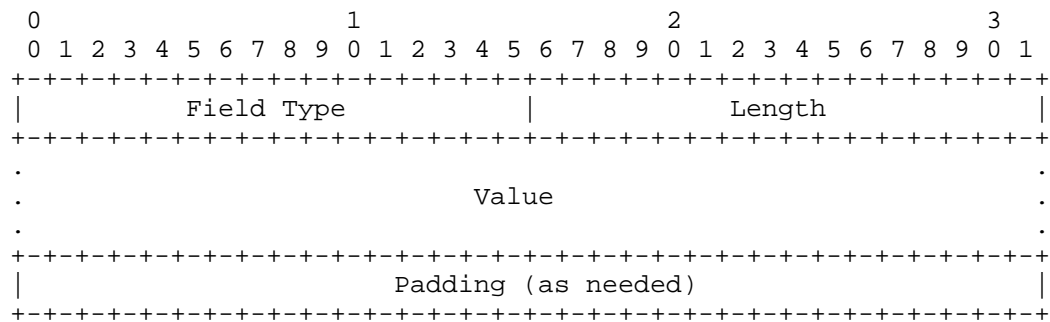


Figure 1 The NTP Extension Field Format

A.4. NTP Extension Field in Autokey

The following paragraph is quoted from Section 10 of [RFC5906].

One or more extension fields follow the NTP packet header and the last followed by the MAC. The extension field parser initializes a pointer to the first octet beyond the NTP packet header and calculates the number of octets remaining to the end of the packet. If the remaining length is 20 (128-bit digest plus 4-octet key ID) or 22 (160-bit digest plus 4-octet key ID), the remaining data are the MAC and parsing is complete. If the remaining length is greater than 22, an extension field is present. If the remaining length is less than 8 or not a multiple of 4, a format error has occurred and the packet is discarded; otherwise, the parser increments the pointer by the extension field length and then uses the same rules as above to determine whether a MAC is present or another extension field.

Authors' Addresses

Tal Mizrahi
 Marvell
 6 Hamada St.
 Yokneam, 20692 Israel

 Email: talmi@marvell.com

Danny Mayer
Network Time Foundation
PO Box 918
Talent OR 97540

Email: mayer@ntp.org

L2VPN Working Group
Internet-Draft
Intended Status: Experimental RFC
Expires: October 10, 2013

Shankar Raman
Balaji Venkat Venkataswami
Gaurav Raina
IIT Madras
Bhargav Bhikkaji
Dell-Force10
April 8, 2013

Securing Model-C Inter-Provider L2 VPNs with Label Hopping and TicToc
draft-mjsraman-l2vpn-vpls-tictoc-label-hop-03

Abstract

In certain models of inter-provider Multi- Protocol Label Switching (MPLS) based Virtual Private Networks (VPNs) spoofing attack against VPN sites is a key concern. For example, MPLS-based VPN inter-provider model "C" for VPLS, or any L2 VPN purpose is not favoured, owing to security concerns in the dataplane, even though it can scale with respect to maintenance of routing state. Since the inner labels associated with VPN sites are not encrypted during transmission, a man-in-the-middle attacker can spoof packets to a specific L2 VPN site. In this paper, we propose a label-hopping technique which uses a set of randomized labels and a method for hopping amongst these labels using the time instant the packet leaves the port from a sending Provider Edge Router. To prevent the attacker from identifying the labels in polynomial time, we also use an additional label. The proposed technique can be applied to other variants of inter-provider MPLS based VPNs where Multi-Protocol exterior-BGP (MP-eBGP) multi-hop is used. As we address a key security concern, we can make a case for the deployment of MPLS based L2 VPN inter-provider model "C". Specifically we use the TicToc based Precision Time Protocol LSP to provide the timing for determining the time instant at which the packet is sent from the remote end Provider Edge Router and hence calculating when it must have left that peer at the Provider Edge Router in the near / receiving end.

This version of the document suggests a better method for gaining more finely granular time slices. This is done by running the PTP LSP between the ASBRs in the ASes that are providing the inter-AS L2VPN service.

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/lid-abstracts.html>

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

Copyright and License Notice

Copyright (c) 2013 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	4
1.1	Terminology	5
2.	Methodology of the proposal	5
2.1	PRE-REQUISITES FOR THE LABEL-HOPPING SCHEME	5
2.1.1	MPLS L2 VPN model "C"	5
2.1.2	PE configuration	6
2.1.3	Control and data-plane flow	6
2.2	LABEL-HOPPING TECHNIQUE	7
2.2.1	Algorithm 1 Control-plane PEne algorithm	8
2.2.2	Algorithm 2 Control-plane PEfa algorithm	10
2.2.3	Algorithm 3 Data-plane PEfa algorithm	11
2.2.4	Algorithm 4 Data-plane PEne algorithm	12

2.2.1 Illustration	13
2.3 SIMULATION AND IMPLEMENTATION	14
2.3.1 Simulation	14
2.3.2 Implementation	14
2.3.3 Running the PTP LSP and label hopping at the ASBRs . . .	15
2.4 CONCLUSION AND FUTURE WORK	16
2.5 ACKNOWLEDGEMENTS	16
3 Security Considerations	17
4 IANA Considerations	17
5 References	17
5.1 Normative References	17
5.2 Informative References	17
Authors' Addresses	18

1 Introduction

Multi-Protocol Label Switching (MPLS) [6] technology uses fixed size labels to forward data packets between routers. By stacking labels, specific customer services such as Layer 2 Virtual Private Networks (L2-VPNs) such as VPLS (Virtual Private Lan Service) based on Border Gateway Protocol (BGP) extensions are widely deployed in the Internet. BGP-based MPLS L2-VPN services are provided either on a single Internet Service Provider (ISP) core or across multiple ISP cores. The latter cases are known as inter-provider MPLS L2-VPNs which are broadly categorized and referred to as models: "A", "B" and "C".

Model "A" uses back-to-back VPN Routing and Forwarding (VRF) connections between Autonomous System Border Routers (ASBRs). Model "B" uses eBGP redistribution of labelled L2 VPN routes from Autonomous Systems (AS) to neighbouring AS. Model "C" uses multi-hop MP-eBGP redistribution of labelled L2 VPN routes and eBGP redistribution of L2 VPN routes from an AS to a neighbouring AS. Model "C" is more scalable for maintaining routing states and hence preferred for deployment in the Internet; refer to [2] for more details. Security issues in MPLS, especially MPLS-based VPNs has attracted attention [1]. The security of model "A" matches the single-AS standard proposed in [9]. Model "B" can be secured well on the control-plane, but on the data-plane the validity of the outer-most label (Label Distribution or Resource Reservation Protocol label) is not checked. This weakness could be exploited to inject crafted packets from inside an MPLS network core. A solution for this problem is proposed in [2]. Model "C" can be secured on the control-plane but has a security weakness on the data-plane. The Autonomous System Border Routers (ASBRs) do not have any VPN information and hence the inner-most label cannot be validated. In this case, the solution used for Model "B" cannot be applied. An attacker can exploit this weakness to send unidirectional packets into the VPN sites connected to the other AS. Therefore, ISPs using model "C" must either trust each other or not deploy it [4].

Control plane security issue in model "C" can be resolved by using IPSec. If IPSec is used in the data-plane then configuring and maintaining key associations could be extremely cumbersome. Even though model "C" is highly scalable for carrying VPN Routing and Forwarding (VRF) L2 VPN routes, the vulnerability of the data-plane renders it unusable. The current recommendation is that model "C" must not be used. In model "C", there are at least two labels for each packet: the Provider Edge (PE) label, which defines the Label Switched Path (LSP) to the egress PE, and the VPN label, which defines the VPN associated with the packet on the PE.

In [5], the authors propose encryption techniques, such as IPSec, for securing the provider edge (PE) of the network. The authors also highlight that the processing capacity could be over-burdened. Further, if an attacker is located at the core of the network, or in the network between the providers that constitute an inter-provider MPLS VPN, then spoofing attacks are possible. The vulnerability of MPLS against spoofing attacks and performance impact of IPSec has been discussed in [3]. If the inner labels that identify packets going towards a L2 VPN site are spoofed, then sensitive information related to services available within the organizational servers can be compromised. As far as we know, there is no scheme available for installing an antispoofing mechanism for these L2 VPN service labels.

This paper outlines a label-hopping technique that helps to alleviate the data-plane security problem in model "C". We propose a scheme that changes the inner L2 VPN labels dynamically based on the time instant the packet is sent from the remote-end PE router. By using a mix of algorithms and randomized labels, we can guard against spoofing and related attacks. The advantage of our scheme is that it can be used wherever Multiprotocol-external BGP (MP-eBGP) multi-hop scenarios arise.

1.1 Terminology

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. Methodology of the proposal

2.1 PRE-REQUISITES FOR THE LABEL-HOPPING SCHEME

In this section, we briefly review the network topology for model "C", the PE configuration and the control-plane exchanges needed for our proposed scheme.

2.1.1 MPLS L2 VPN model "C"

The reference MPLS-eBGP based L2 VPN network for model "C" as described in [11] is shown in Figure 1, which also shows the control plane exchanges. The near-end PE (PEne) and far-end PE (PEfa) are connected through the inter-provider MPLS core. The VPN connectivity is established through a set of routers from different Autonomous Systems (AS) and their ASBRs. In the L2 VPN, MP-eBGP updates are exchanged for a set of MAC based Forward Equivalence Classes (FECs). These FECs, which have to be protected, originate from the MAC addresses / FECs behind PEne in a L2 VPN site or a set of L2 VPN

sites.

2.1.2 PE configuration

Various configurations are needed on the PEs to implement the label hopping scheme. A set of "m" algorithms that generate collision-free labels (universal hashing algorithms) are initially implemented in the PEs. Each algorithm is mapped to an index $A = (a_1; a_2; \dots a_m)$ where $m \geq 1$. The bit-selection pattern used by the PEs for generating the additional label is also configured. PEne must be configured for a FEC or a set of FECs represented by an aggregate label (per VRF label) which will use the label-hopping scheme. For each FEC or a set of FECs, a set of valid labels used for hopping, $K = (k_1; k_2; k_3; \dots k_n)$ where $n \geq 1$ and, $k_i \neq k_j$ if $i \neq j$, is configured in PEne. For the set of labels K time slices $TS = (TS_1; TS_2; TS_3 \dots TS_n)$ are also exchanged. These time slices can be periodically changed and a new set of TS ranging from TS_1 to TS_n can be exchanged after a time duration $TS_Exchange_Interval$ which itself can be randomized from time to time. In the case of bi-directional security, the roles of the PEs can be reversed. In addition to these data sets a random seed is also exchanged. This Random Seed which we will henceforth as $Rseed$ is used to generate the label for the next time slot.

2.1.3 Control and data-plane flow

Initially, set K , set TS and the bit-selection pattern used by the PEs are exchanged securely over the control-plane. Optionally an index from A , representing a hash-algorithm, could also be exchanged. We propose that only the index is exchanged between the PEs, as it enhances the security, for two reasons. First, the algorithm itself is masked from the attacker. Second, the algorithm can be changed frequently, and it would be difficult for the attacker to identify the final mapping that generates the label to be used for a packet. Figure 1 depicts this unidirectional exchange from PEne to PEfa.

The control plane exchanges also involve a-priori constructing a Precision Time Protocol (PTP) LSP for deriving the clock at the PEne and PEfa for a forwarding direction. For the reverse direction another PTP LSP can be constructed as well. In the example that we illustrate we discuss about only a single forwarding direction. The PTP LSP port assigned for a forwarding direction is tied in with the configuration that goes into the inter-PEne-PEfa exchanges to setup the labelling control plane. So each pair of PEne and PEfa knows which PTP port and corresponding PTP LSP as per [12] to be used for the traffic. The PTP LSP is intended for providing the clocking between a pair of PEne and PEfa. The clock / timestamp derived from this PTP LSP is used in the data plane operation to determine which

label is valid at that time instant as will be seen in the Algorithms provided below.

Once the secure control-plane exchanges are completed, we apply the label-hopping technique, and PEfa forwards the labelled traffic towards PEne through the intermediate routers using the label-stacking technique (Figure 2). The stacked labels along with the payload are transferred between the AS and ASBRs before they reach PEne. Using the label-hopping algorithm PEne verifies the integrity of labels. Upon validation, PEne uses the label information to forward the packets to the appropriate L2 VPN service instance or site. This data-plane exchange from PEfa and PEne is depicted in Figure 3. We now present the label-hopping scheme.

2.2 LABEL-HOPPING TECHNIQUE

In this section, we describe the label-hopping technique and discuss some implementation aspects. Once a data packet destined to the PEne arrives at the PEfa (a) a first-label is chosen using set K and set TS, and the random seed Rseed, and a first-label selected. Next (b) a selected number of bytes from the payload is chosen as input to the hashing algorithm. The hash-digest obtained as a result is used to obtain the additional label for the packet. The agreed bit-selection pattern is then applied on the hash-digest to obtain an additional label, which is then concatenated with the first label. Once PEne receives these packets it verifies both the labels.

The implementation steps for the control-plane at the PEne and PEfa are given by Algorithms 1 and 2. The implementation steps for the data-plane at the PEfa and PEne are given by Algorithms 3 and 4.

2.2.1 Algorithm 1 Control-plane PEne algorithm

Require:

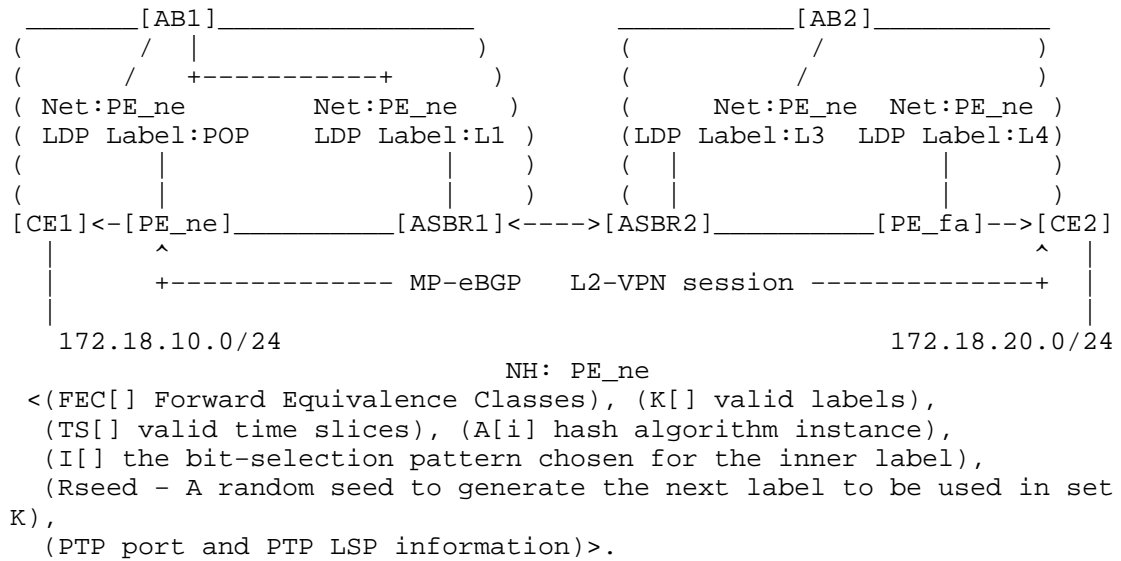
- * FEC[] Forward Equivalence Classes,
- * K[] valid labels,
- * TS[] valid time slices,
- * A[i] hash algorithm instance,
- * I[] the bit-selection pattern chosen for the inner label.
- * Random seed "Rseed" which is used for generating the index into set K (set of labels).
- * PTP port and PTP LSP information

Begin

```
packet = makepacket(FEC,K, TS, A[i], I, Rseed);  
CP-SendPacket(PEfa, MP-eBGP, packet);  
End
```

Note: The values in K need not be contiguous and can be randomly chosen from a pool of labels to remove coherence in the label space. Also the algorithms used could be either vendor dependent or a set of standard algorithms mapped the same way by the PEne and PEfa. If the two PEs involved are from different vendors we assume that a set of standard algorithms are used.

Note: Also the values in set TS should be of a coarse granularity of seconds recommended to be higher than 2 seconds.



Exchange all details as per Algorithm 1.

Figure 1: Control-plane exchanges for model C [11]

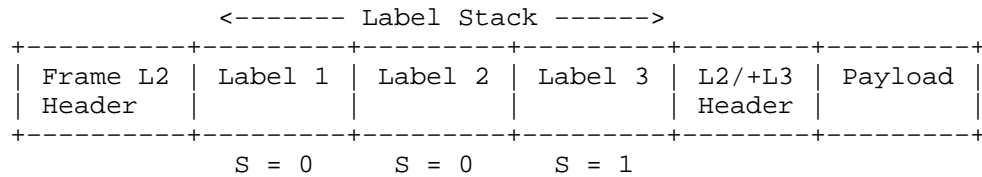


Figure 2: Label stack using scheme outlined for Model "C"

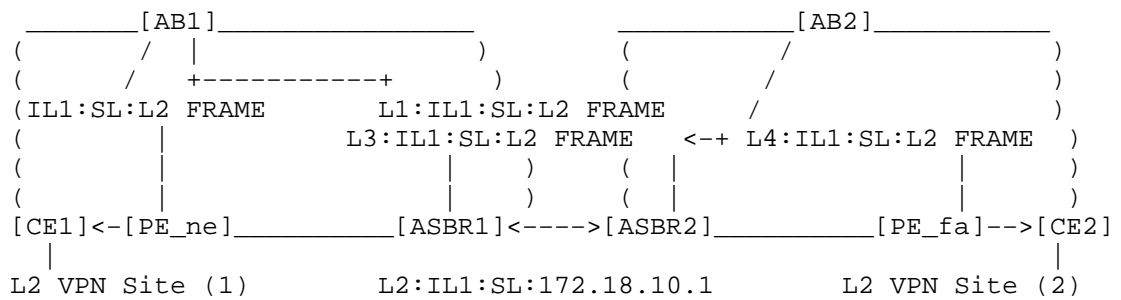


Figure 3: Data-plane flow for model C [11]

2.2.2 Algorithm 2 Control-plane PEfa algorithm

```
Require: None
Begin
packet = CP-ReceivePacket(PEnet); // from PEnet
FEC[] = ExtractFEC(packet); // extract FECs
K[] = ExtractLabels(packet); // extract the labels
TS[] = ExtractTimeSlices(packet); // extract the time slices
Rseed = ExtractRandomSeed(packet); // extract the Rseed value.
selectHashAlgorithm(A[i]); // hash algorithm to use
RecordValues(FEC); // information for PEfa
RecordValues(K);
RecordValues(TS);
RecordValues(I); // bit-selection pattern to be used
RecordValue(Rseed);
End
```

2.2.3 Algorithm 3 Data-plane PEfa algorithm

```

Require: None

Begin
  Initialization :

  One Time Init :

  BeginInit

  CurrentTimeSliceIndex = 0;

  CurrentMasterClock = PTP LSP Master Clock Timestamp;

  CurrentTimeInstant = CurrentMasterClock;

  NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];

  EndInit

  packet = DP-ReceivePacket(Interface);
  match = CheckFEC(packet); // Is the algorithm enabled?
  if match == 0 then
    return; // no match
  end if
  hash-digest = calculateHash(A[i],packet);
  if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
    // do nothing;
  else
    CurrentTimeSliceIndex++;
    if CurrentTimeSliceIndex == n then // check to wrap around
      CurrentTimeSliceIndex = 0;
    end if
    CurrentTimeInstant = NextTimeInstant;
    NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
  end if
  first-label = K[GenerateRandom(Rseed) MOD n(K)];
  end if
  additional-label = process(hash-digest,I)
  DP-SendPacket(PEnet, first-label, additional-label, packet);
End

```

2.2.4 Algorithm 4 Data-plane PEne algorithm

```

Require: None
Initialization :
One Time Init :

BeginInit
CurrentTimeSliceIndex = 0;
CurrentMasterClock = PTP LSP Clock Timestamp;
CurrentTimeInstant = CurrentMasterClock;
NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];
EndInit

Begin
packet = DP-ReceivePacket(Interface);
match = CheckFEC(packet);
if match == 0 then
    return; //no match
end if

label-in-packet=extractPacket(packet, LABEL);
inner-label=extractPacket(packet, INNER-LABEL);
hash-digest=calculateHash(A[i],packet);
if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
    // do nothing;
else
    CurrentTimeSliceIndex++;
    // Save the old RseedIndex into set K
    OldRseedIndex = RseedIndex;
    RseedIndex = (GenerateRandom(Rseed) MOD n(K));
    NextRseedIndex =
        LookAheadRseedIndex(GenerateRandom(Rseed) MOD n(K));
    RollbackRseed(Rseed by 1);
    if CurrentTimeSliceIndex == n then // check to wrap around
        CurrentTimeSliceIndex = 0;
    end if
    CurrentTimeInstant = NextTimeInstant;
    NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
end if
// Check if label used before in the previous | current or future
// time slot can be used
// Check with OldRseedIndex, RseedIndex and NextRseedIndex
first-label-range = K[RseedIndex (+or- 1)];
additional-label = process(hash-digest,I)
if label-in-packet ! in first-label-range then
    error(); return;
end if

```

```

if inner-label != additional-label then
    error(); return;
end if
DP-SendPacket(CE1, NULL, NULL, packet);
End

```

Here configured seconds could be a fraction as well.

In order to avoid too many processing cycles in the line cards of PEne and PEfa, the hash- digest is calculated over a predefined size of the payload. An additional inner label is further added to enhance protection against spoofing attacks. With an increased label size, an attacker spends more than polynomial time to guess the VPN instance label for the site behind PEne. There could be two hash-digests that generate the same label. In this case, the two hash-digests is differentiated using the additional label. Collisions can be avoided by re-hashing or any other suitable techniques that are proposed in the literature [8]. If collisions exceed a certain number, then Algorithms 1 and 2 can be executed with a set of new labels.

Note :

It is to be noted that the change in the algorithm to randomly pick up a label for the next time slot will help in avoiding man-in-the-middle attackers from synchronizing with the time slots and the labels which in the previous version of the algorithm was predictable if a large number of packets were observed. The Random seed agreed upon will generate in lock step with the time slots at both the PEfa and PEne, the correct label to be used and that will throw off the attacker from synchronizing with such label changes. Thus even replay attacks may be harder to attempt in such a case.

2.2.1 Illustration

We now briefly illustrate the label-hopping scheme. In Figure 1, using Algorithms 1 and 2, a set of labels are forwarded from PEne to PEfa. The roles of PEne and PEfa are interchanged for reverse traffic. Figure 2 shows a packet from the data-plane for model "C", with the proposed scheme. In the figure, "Label 1" refers to the outermost label, while "Label 2" refers to the label generated from the set K and set TS and "Label 3" refers to an additional label generated as in Algorithm 3. This additional label has bottom of stack bit (denoted by S in Figure 2) set. These labels are stacked immediately onto the packet and the path labels for routing the packets to appropriate intermediary PEs are added. Figure 3 also shows these path labels used by the data packet to reach PEne. When the packet passes through the core of an intermediary AS involved in model "C", or through the network connecting the intermediary AS, the

intruder or the attacker has the capability to inspect the labels and the payload. However, the proposed scheme prevents the attacker from guessing the right combination of the labels. We can increase the size of the additional inner-labels thereby reducing threats from polynomial time attacks.

2.3 SIMULATION AND IMPLEMENTATION

In this section, we present the preliminary simulation results on performance, comparing the label-hopping technique with deep packet inspection where we encrypt and decrypt the complete packet. We also briefly highlight some implementation issues.

2.3.1 Simulation

Implementing the label-hopping scheme for all set of FECs belonging to any or all VPN service instances may cause throughput degradation. This is because the hashdigest computation and derivation of the inner-label / additional inner label calculation can be computation intensive. We therefore compared our technique by choosing a part of the payload as input to our hashing algorithm. We simulated our algorithm on a 2.5 GHz processor Intel dual processor quad core machine. We compared the performance of the label-hopping technique with a deep packet inspection technique where the complete packet was encrypted before transmission and decrypted on reception. These simulation figures indicate that we were able to process 10 million packets per second when we used 64-byte for hashing on a payload of size 1024 bytes. For a hash using 128-byte, we were able to process about 6.3 million packets per second. However with a deep packet inspection where we encrypted and decrypted the complete packet, we were able to process only about 1 million packets per second. In cases where performance becomes a bottleneck, this label-hopping scheme can be applied to specific traffic which are mission-critical, sensitive and most likely need to be protected as they travel from the PEfa to the PEna. Selective application of this service which could be offered as a premium for a selected set of FECs is a suitable option, thereby protecting the traffic of organizations that are paranoid about the integrity of the switched traffic into their VPN sites.

2.3.2 Implementation

One of the concerns in the scheme is the use of payload for generating the random inner label / additional label. If the payload does not vary between two packets then the control-plane exchanges have to be renegotiated with a different algorithm to be used for the hashing for the subsequent packets. The other concern in the scheme is to tackle the problem of fragmentation that can occur along the

path from PEfa to PEne. We can fragment the packet at PEfa and ensure that the size of the packet is fixed before transmission. We could also employ the Path Maximum Transfer Unit (Path-MTU) discovery process so that packets do not get split into multiple fragments. If packets are fragmented this scheme fails. However, networks usually employ the Path-MTU discovery process to prevent fragmentation and hence this problem may not occur.

2.3.3 Running the PTP LSP and label hopping at the ASBRs

The ASes participating in the inter-AS L2VPN Option-C type service connect with each other using ASBRs that connect one AS to another.

It would be prudent to run the PTP LSP and the label-hopping algorithm between the ASBRs instead of between the PEs. Since these ASBRs are usually one-hop away from each other or in the worst case a couple of hops away, the granularity of the time slices can be a lot more finer than when running between the PEs. At more granular time slices it will be even harder for an attacker to pump in packets that utilize the slack of + or - microseconds or milliseconds configured in Algorithm 4.

Hence spoofing and replay attacks are less likely to succeed.

To make it clear the innermost label which is the hash digest computed on the first 128 or 64 byte portion of the payload which is binary anded with an arbitrary bit pattern known to both PEs in the topology , serves as an added binary pattern which has to be guessed by the intruder intending to spoof the packet into the VPN's PE onto the CE.

Thus the effective label space that has to be guessed by the intruder is the label for that time slice and the binary pattern computed on the payload (result of the hash-digest ANDed with the arbitrary bit pattern).

This makes it essentially a 40 bit label space. The hash-digest was not intended to be a ICV. It could serve as an ICV as well.

Since the binary pattern exchanged through the control plane is not known to the intruder, and the hash algorithm used is not known to the intruder (unless of course both of them are compromised in the control plane exchange which is of course secure) the resulting innermost label extends the label space to 40 bits (including the label for that time slice) that has to be guessed.

As to whether there might be a flood of replay packets with the + or - 1 time slice being in place, the previous label used would be known

but the one after the current time slice would be hard to guess owing to the random number generation function being used to determine which the next label should be. It should be possible to jam for that time slice with the same packet with the 2 labels (previous and current) for that time slice being repeated again and again. This is solved by finely granularizing the time slices to microseconds or milliseconds. This is especially the case if the scheme is run between the ASBRs and not between the PEs.

2.4 CONCLUSION AND FUTURE WORK

In this paper, we proposed a label-hopping scheme for inter-provider BGP-based MPLS L2 VPNs that employ MPE-BGP multi-hop control-plane exchanges. In such an environment, without label-hopping, the data-plane is subject to spoofing attacks.

The technique proposed uses a time-based label hopping scheme in addition to the use of the payload to generate an inner label to prevent attackers from easily deciphering labels and their respective VPNs. The scheme is less computationally intensive than encryption-based methods. It prevents the spoofed packets from getting into a VPN site even if the attacker is in the core or at an intervening link between ISPs. In our scheme, we chose the time instant that the packet leaves the first Provider Edge on the far end and this time instant serves as the variable component that the attacker cannot decipher. This requires the use of time synchronization mechanism. This is provided by the PTP LSP constructed for this purpose.

2.5 ACKNOWLEDGEMENTS

The authors would like to acknowledge the UK EP-SRC Digital Economy Programme and the Government of India Department of Science and Technology (DST) for funding given to the IU-ATC. The authors would also like to thank Chandrasekhar.R and Narayana Swamy for his review and valuable comments during the writing of this draft.

3 Security Considerations

The main objective of this proposal is to secure the Inter-Provider MPLS L2 VPN Model-C data plane by preventing spoofing attacks and other unidirectional attacks against the customer site in this model. The suggestions and algorithms provided will mitigate these attacks to a large extent. The attacker will have many barriers to break through before he/she can successfully mount an attack against the customer site in this model with these algorithms implemented. The availability of TicToc as a method of clocking helps a great deal in this direction.

4 IANA Considerations

Appropriate IANA indicators would have to be provided to exchange the set of values that Algorithm 1 outlines in order to implement this scheme.

5 References

5.1 Normative References

5.2 Informative References

- [1] S. Alouneh, A. En-Nouaary and A. Agarwal, "MPLS security: an approach for unicast and multicast environments", Annals of Telecommunications, Springer, vol. 64, no. 5, June 2009, pp. 391-400, doi:10.1007/s12243-009-0089-y.
- [2] M. H. Behringer and M. J. Morrow, "MPLS VPN security", Cisco Press, June 2005, ISBN-10: 1587051834.
- [3] B. Daugherty and C. Metz, "Multiprotocol Label Switching and IP, Part 1, MPLS VPNS over IP Tunnels", IEEE Internet Computing, May-June 2005, pp. 68-72, doi: 10.1109/MIC.2005.61.
- [4] L. Fang, N. Bitar, J. L. Le Roux and J. Miles, "Interprovider IP-MPLS services: requirements, implementations, and challenges", IEEE Communications Magazine, vol. 43, no. 6, June 2005, pp. 119-128, doi: 10.1109/MCOM.2005.1452840.
- [5] C. Lin and W. Guowei, "Security research of VPN

technology based on MPLS", Proceedings of the Third International Symposium on Computer Science and Computational Technology (ISCSCT 10), August 2010, pp. 168-170, ISBN- 13:9789525726107.

[6] Y. Rekhter, B. Davie, E. Rosen, G. Swallow, D. Farinacci and D. Katz, "Tag switching architecture overview", Proceedings of the IEEE, vol. 85, no. 12, December 1997, pp. 1973-1983, doi:10.1109/5.650179.

[7] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, Standard Track, February, 2006.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to algorithms", 3rd edition, MIT Press, September 2009, ISBN-10:0262033844.

[9] C. Semeria, "RFC 2547bis: BGP/MPLS VPN fundamentals", Juniper Networks white paper, March 2001.

[10] Advance MPLS VPN Security Tutorials [Online], Available:
"http://etutorials.org/Networking/MPLS+VPN+security/Part+II+Advanced+MPLS+VPN+Security+Issues/", [Accessed: 10th December 2011]

[11] Inter-provider MPLS VPN models [Online], Available:
"http://mpls-configuration-on-cisco-iossoftware.org.ua/1587051990/ ch07levlsec4.html", [Accessed 10th December 2011]

[12] Davari.S et.al, Transporting PTP messages (1588) over MPLS networks, "http://datatracker.ietf.org/doc/draft-ietf-tictoc-1588overmpls/?include_text=1", Work in Progress, October 2011.

Authors' Addresses

Shankar Raman
Department of Computer Science and Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: mjsraman@cse.iitm.ac.in

Balaji Venkat Venkataswami
Department of Electrical Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: balajivenkat299@gmail.com

Prof.Gaurav Raina
Department of Electrical Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: gaurav@ee.iitm.ac.in

Bhargav Bhikkaji
Dell-Force10
350 Holger Way
San Jose, CA
USA

Email: Bhargav_Bhikkaji@dell.com

L3VPN Working Group
Internet-Draft
Intended Status: Experimental RFC
Expires: October 10, 2013

Shankar Raman
Balaji Venkat Venkataswami
Gaurav Raina
IIT, Madras
April 8, 2013

Securing Model-C Inter-Provider VPNs with Label Hopping and TicToc
draft-mjsraman-l3vpn-tictoc-label-hop-03

Abstract

In certain models of inter-provider Multi- Protocol Label Switching (MPLS) based Virtual Private Networks (VPNs) spoofing attack against VPN sites is a key concern. For example, MPLS-based VPN inter-provider model "C" is not favoured, owing to security concerns in the dataplane, even though it can scale with respect to maintenance of routing state. Since the inner labels associated with VPN sites are not encrypted during transmission, a man-in-the-middle attacker can spoof packets to a specific VPN site. In this paper, we propose a label-hopping technique which uses a set of randomized labels and a method for hopping amongst these labels using the time instant the packet leaves the port from a sending Provider Edge Router. To prevent the attacker from identifying the labels in polynomial time, we also use an additional label. The proposed technique can be applied to other variants of inter-provider MPLS based VPNs where Multi-Protocol exterior-BGP (MP-eBGP) multi-hop is used. As we address a key security concern, we can make a case for the deployment of MPLS based VPN inter-provider model "C". Specifically we use the TicToc based Precision Time Protocol LSP to provide the timing for determining the time instant at which the packet is sent from the remote end Provider Edge Router and hence calculating when it must have left that peer at the Provider Edge Router in the near / receiving end.

This version of the document suggests a better method for gaining more finely granular time slices. This is done by running the PTP LSP between the ASBRs in the ASes that are providing the inter-AS L3VPN service.

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/lid-abstracts.html>

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

Copyright and License Notice

Copyright (c) 2013 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	4
1.1	Terminology	5
2.	Methodology of the proposal	5
2.1	PRE-REQUISITES FOR THE LABEL-HOPPING SCHEME	5
2.1.1	MPLS VPN model "C"	5
2.1.2	PE configuration	6
2.1.3	Control and data-plane flow	6
2.2	LABEL-HOPPING TECHNIQUE	7
2.2.1	Algorithm 1 Control-plane PEne algorithm	8
2.2.2	Algorithm 2 Control-plane PEfa algorithm	10
2.2.3	Algorithm 3 Data-plane PEfa algorithm	11
2.2.4	Algorithm 4 Data-plane PEne algorithm	12
2.2.1	Illustration	13
2.3	SIMULATION AND IMPLEMENTATION	14

2.3.1 Simulation	14
2.3.2 Implementation	14
2.3.3 Running the PTP LSP and label hopping at the ASBRs . . .	15
2.4 CONCLUSION AND FUTURE WORK	16
2.5 ACKNOWLEDGEMENTS	16
3 Security Considerations	17
4 IANA Considerations	17
5 References	17
5.1 Normative References	17
5.2 Informative References	17
Authors' Addresses	19

1 Introduction

Multi-Protocol Label Switching (MPLS) [6] technology uses fixed size labels to forward data packets between routers. By stacking labels, specific customer services such as Layer 3 Virtual Private Networks (L3-VPNs) based on Border Gateway Protocol (BGP) extensions are widely deployed in the Internet. BGP-based MPLS L3-VPN services are provided either on a single Internet Service Provider (ISP) core or across multiple ISP cores. The latter cases are known as inter-provider MPLS VPNs which are broadly categorized and referred to as models: "A", "B" and "C" [10].

Model "A" uses back-to-back VPN Routing and Forwarding (VRF) connections between Autonomous System Border Routers (ASBRs). Model "B" uses eBGP redistribution of labelled VPN-IPv4 routes from Autonomous Systems (AS) to neighbouring AS. Model "C" uses multi-hop MP-eBGP redistribution of labelled VPN-IPv4 routes and eBGP redistribution of IPv4 routes from an AS to a neighbouring AS. Model "C" is more scalable for maintaining routing states and hence preferred for deployment in the Internet; refer to [2] for more details. Security issues in MPLS, especially MPLS-based VPNs has attracted attention [1]. The security of model "A" matches the single-AS standard proposed in [9]. Model "B" can be secured well on the control-plane, but on the data-plane the validity of the outer-most label (Label Distribution or Resource Reservation Protocol label) is not checked. This weakness could be exploited to inject crafted packets from inside an MPLS network core. A solution for this problem is proposed in [2]. Model "C" can be secured on the control-plane but has a security weakness on the data-plane. The Autonomous System Border Routers (ASBRs) do not have any VPN information and hence the inner-most label cannot be validated. In this case, the solution used for Model "B" cannot be applied. An attacker can exploit this weakness to send unidirectional packets into the VPN sites connected to the other AS. Therefore, ISPs using model "C" must either trust each other or not deploy it [4].

Control plane security issue in model "C" can be resolved by using IPSec. If IPSec is used in the data-plane then configuring and maintaining key associations could be extremely cumbersome. Even though model "C" is highly scalable for carrying VPN Routing and Forwarding (VRF) routes, the vulnerability of the data-plane renders it unusable. The current recommendation is that model "C" must not be used. A simple solution to this problem is to filter all IP traffic with the exception of the required eBGP peering between the ASBRs, thereby preventing a large number of potential IP traffic-related attacks. However, controlling labelled packets is difficult. In model "C", there are at least two labels for each packet: the Provider Edge (PE) label, which defines the Label Switched Path (LSP) to the egress

PE, and the VPN label, which defines the VPN associated with the packet on the PE.

In [5], the authors propose encryption techniques, such as IPSec, for securing the provider edge (PE) of the network. The authors also highlight that the processing capacity could be over-burdened. Further, if an attacker is located at the core of the network, or in the network between the providers that constitute an inter-provider MPLS VPN, then spoofing attacks are possible. The vulnerability of MPLS against spoofing attacks and performance impact of IPSec has been discussed in [3]. If the inner labels that identify packets going towards a L3 VPN site are spoofed, then sensitive information related to services available within the organizational servers can be compromised. As far as we know, there is no scheme available for installing an antispoofing mechanism for these VPN service labels.

This paper outlines a label-hopping technique that helps to alleviate the data-plane security problem in model "C". We propose a scheme that changes the inner VPN labels dynamically based on the time instant the packet is sent from the remote-end PE router. By using a mix of algorithms and randomized labels, we can guard against spoofing and related attacks. The advantage of our scheme is that it can be used wherever Multiprotocol-external BGP (MP-eBGP) multi-hop scenarios arise.

1.1 Terminology

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. Methodology of the proposal

2.1 PRE-REQUISITES FOR THE LABEL-HOPPING SCHEME

In this section, we briefly review the network topology for model "C", the PE configuration and the control-plane exchanges needed for our proposed scheme.

2.1.1 MPLS VPN model "C"

The reference MPLS-eBGP based VPN network for model "C" as described in [11] is shown in Figure 1, which also shows the control plane exchanges. The near-end PE (PEne) and far-end PE (PEfa) are connected through the inter-provider MPLS core. The VPN connectivity is established through a set of routers from different Autonomous Systems (AS) and their ASBRs. In the VPN, MP-eBGP updates are

exchanged for a set of Forward Equivalence Classes (FECs). These FECs, which have to be protected, originate from the prefixes behind PEne in a VPN site or a set of VPN sites.

2.1.2 PE configuration

Various configurations are needed on the PEs to implement the label hopping scheme. A set of "m" algorithms that generate collision-free labels (universal hashing algorithms) are initially implemented in the PEs. Each algorithm is mapped to an index $A = (a_1; a_2; \dots a_m)$ where $m \geq 1$. The bit-selection pattern used by the PEs for generating the additional label is also configured. PEne must be configured for a FEC or a set of FECs represented by an aggregate label (per VRF label) which will use the label-hopping scheme. For each FEC or a set of FECs, a set of valid labels used for hopping, $K = (k_1; k_2; k_3; \dots k_n)$ where $n \geq 1$ and, $k_i \neq k_j$ if $i \neq j$, is configured in PEne. For the set of labels K time slices $TS = (TS_1; TS_2; TS_3 \dots TS_n)$ are also exchanged. These time slices can be periodically changed and a new set of TS ranging from TS_1 to TS_n can be exchanged after a time duration TS_Exchange_Interval which itself can be randomized from time to time. In the case of bi-directional security, the roles of the PEs can be reversed. In addition to these data sets a random seed is also exchanged. This Random Seed which we will henceforth as R_{seed} is used to generate the label for the next time slot.

2.1.3 Control and data-plane flow

Initially, set K , set TS and the bit-selection pattern used by the PEs are exchanged securely over the control-plane. Optionally an index from A , representing a hash-algorithm, could also be exchanged. We propose that only the index is exchanged between the PEs, as it enhances the security, for two reasons. First, the algorithm itself is masked from the attacker. Second, the algorithm can be changed frequently, and it would be difficult for the attacker to identify the final mapping that generates the label to be used for a packet. Figure 1 depicts this unidirectional exchange from PEne to PEfa.

The control plane exchanges also involve a-priori constructing a Precision Time Protocol (PTP) LSP for deriving the clock at the PEne and PEfa for a forwarding direction. For the reverse direction another PTP LSP can be constructed as well. In the example that we illustrate we discuss about only a single forwarding direction. The PTP LSP port assigned for a forwarding direction is tied in with the configuration that goes into the inter-PEne-PEfa exchanges to setup the labelling control plane. So each pair of PEne and PEfa knows which PTP port and corresponding PTP LSP as per [12] to be used for the traffic. The PTP LSP is intended for providing the clocking

between a pair of PEne and PEfa. The clock / timestamp derived from this PTP LSP is used in the data plane operation to determine which label is valid at that time instant as will be seen in the Algorithms provided below.

Once the secure control-plane exchanges are completed, we apply the label-hopping technique, and PEfa forwards the labelled traffic towards PEne through the intermediate routers using the label-stacking technique (Figure 2). The stacked labels along with the payload are transferred between the AS and ASBRs before they reach PEne. Using the label-hopping algorithm PEne verifies the integrity of labels. Upon validation, PEne uses the label information to forward the packets to the appropriate VPN service instance or site. This data-plane exchange from PEfa and PEne is depicted in Figure 3. We now present the label-hopping scheme.

2.2 LABEL-HOPPING TECHNIQUE

In this section, we describe the label-hopping technique and discuss some implementation aspects. Once a data packet destined to the PEne arrives at the PEfa (a) a first-label is chosen using set K and set TS, and the random seed Rseed, and a first-label selected. Next (b) a selected number of bytes from the payload is chosen as input to the hashing algorithm. The hash-digest obtained as a result is used to obtain the additional label for the packet. The agreed bit-selection pattern is then applied on the hash-digest to obtain an additional label, which is then concatenated with the first label. Once PEne receives these packets it verifies both the labels.

The implementation steps for the control-plane at the PEne and PEfa are given by Algorithms 1 and 2. The implementation steps for the data-plane at the PEfa and PEne are given by Algorithms 3 and 4.

2.2.1 Algorithm 1 Control-plane PEne algorithm

Require:

- * FEC[] Forward Equivalence Classes,
- * K[] valid labels,
- * TS[] valid time slices,
- * Random seed "Rseed" which is used for generating the index into set K (set of labels).
- * A[i] hash algorithm instance,
- * I[] the bit-selection pattern chosen for the inner label.
- * PTP port and PTP LSP information

Begin

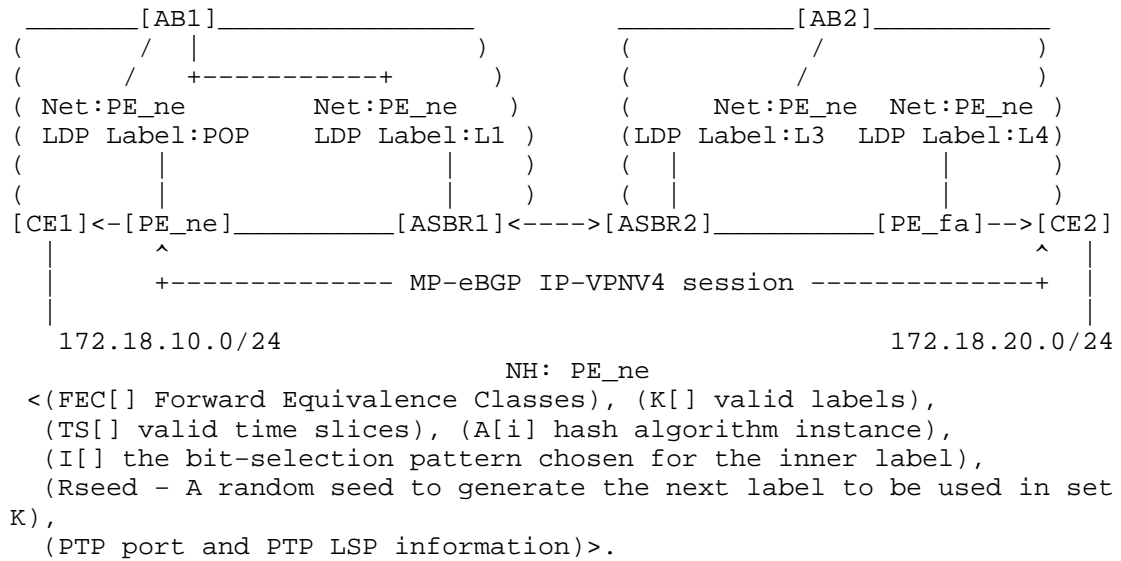
```
packet = makepacket(FEC,K, TS, A[i], I, Rseed);
```

```
CP-SendPacket(PEfa, MP-eBGP, packet);
```

End

Note: The values in K need not be contiguous and can be randomly chosen from a pool of labels to remove coherence in the label space. Also the algorithms used could be either vendor dependent or a set of standard algorithms mapped the same way by the PEne and PEfa. If the two PEs involved are from different vendors we assume that a set of standard algorithms are used.

Note: Also the values in set TS should be of a coarse granularity of seconds recommended to be higher than 2 seconds.



Exchange all details as per Algorithm 1.

Figure 1: Control-plane exchanges for model C [11]

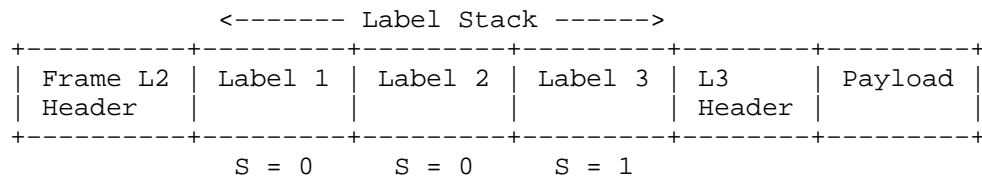


Figure 2: Label stack using scheme outlined for Model "C"

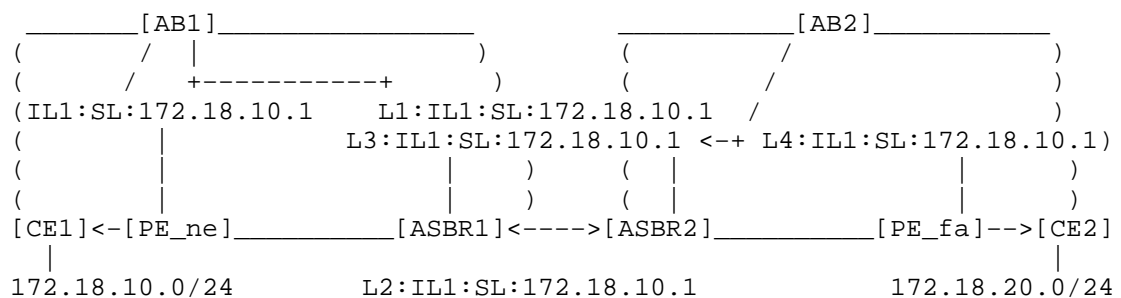


Figure 3: Data-plane flow for model C [11]

2.2.2 Algorithm 2 Control-plane PEfa algorithm

```
Require: None
Begin
packet = CP-ReceivePacket(PEn); // from PEn
FEC[] = ExtractFEC(packet); // extract FECs
K[] = ExtractLabels(packet); // extract the labels
TS[] = ExtractTimeSlices(packet); // extract the time slices
Rseed = ExtractRandomSeed(packet); // extract the Rseed value.
selectHashAlgorithm(A[i]); // hash algorithm to use
RecordValues(FEC); // information for PEfa
RecordValues(K);
RecordValues(TS);
RecordValues(I); // bit-selection pattern to be used
RecordValue(Rseed);
End
```


2.2.3 Algorithm 3 Data-plane PEfa algorithm

```

Require: None

Begin
  Initialization :

  One Time Init :

  BeginInit

  CurrentTimeSliceIndex = 0;

  CurrentMasterClock = PTP LSP Master Clock Timestamp;

  CurrentTimeInstant = CurrentMasterClock;

  NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];

  EndInit

  packet = DP-ReceivePacket(Interface);
  match = CheckFEC(packet); // Is the algorithm enabled?
  if match == 0 then
    return; // no match
  end if
  hash-digest = calculateHash(A[i],packet);
  if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
    // do nothing;
  else
    CurrentTimeSliceIndex++;
    if CurrentTimeSliceIndex == n then // check to wrap around
      CurrentTimeSliceIndex = 0;
    end if
    CurrentTimeInstant = NextTimeInstant;
    NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
  end if
  first-label = K[GenerateRandom(Rseed) MOD n(K)];
  end if
  additional-label = process(hash-digest,I)
  DP-SendPacket(PEnet, first-label, additional-label, packet);
End

```

2.2.4 Algorithm 4 Data-plane PEne algorithm

```

Require: None
Initialization :
One Time Init :

BeginInit
CurrentTimeSliceIndex = 0;
CurrentMasterClock = PTP LSP Clock Timestamp;
CurrentTimeInstant = CurrentMasterClock;
NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];
EndInit

Begin
packet = DP-ReceivePacket(Interface);
match = CheckFEC(packet);
if match == 0 then
    return; //no match
end if

label-in-packet=extractPacket(packet, LABEL);
inner-label=extractPacket(packet, INNER-LABEL);
hash-digest=calculateHash(A[i],packet);
if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
    // do nothing;
else
    CurrentTimeSliceIndex++;
    // Save the old RseedIndex into set K
    OldRseedIndex = RseedIndex;
    RseedIndex = (GenerateRandom(Rseed) MOD n(K));
    NextRseedIndex =
        LookAheadRseedIndex(GenerateRandom(Rseed) MOD n(K));
    RollbackRseed(Rseed by 1);
    if CurrentTimeSliceIndex == n then // check to wrap around
        CurrentTimeSliceIndex = 0;
    end if
    CurrentTimeInstant = NextTimeInstant;
    NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
end if
// Check if label used before in the previous | current or future
// time slot can be used
// Check with OldRseedIndex, RseedIndex and NextRseedIndex
first-label-range = K[RseedIndex (+or- 1)];
additional-label = process(hash-digest,I)
if label-in-packet ! in first-label-range then
    error(); return;
end if

```

```

if inner-label != additional-label then
    error(); return;
end if
DP-SendPacket(CE1, NULL, NULL, packet);
End

```

Here configured seconds could be a fraction as well.

In order to avoid too many processing cycles in the line cards of PEne and PEfa, the hash- digest is calculated over a predefined size of the payload. An additional inner label is further added to enhance protection against spoofing attacks. With an increased label size, an attacker spends more than polynomial time to guess the VPN instance label for the site behind PEne. There could be two hash-digests that generate the same label. In this case, the two hash-digests is differentiated using the additional label. Collisions can be avoided by re-hashing or any other suitable techniques that are proposed in the literature [8]. If collisions exceed a certain number, then Algorithms 1 and 2 can be executed with a set of new labels.

Note :

It is to be noted that the change in the algorithm to randomly pick up a label for the next time slot will help in avoiding man-in-the-middle attackers from synchronizing with the time slots and the labels which in the previous version of the algorithm was predictable if a large number of packets were observed. The Random seed agreed upon will generate in lock step with the time slots at both the PEfa and PEne, the correct label to be used and that will throw off the attacker from synchronizing with such label changes. Thus even replay attacks may be harder to attempt in such a case.

2.2.1 Illustration

We now briefly illustrate the label-hopping scheme. In Figure 1, using Algorithms 1 and 2, a set of labels are forwarded from PEne to PEfa. The roles of PEne and PEfa are interchanged for reverse traffic. Figure 2 shows a packet from the data-plane for model "C", with the proposed scheme. In the figure, "Label 1" refers to the outermost label, while "Label 2" refers to the label generated from the set K and set TS and "Label 3" refers to an additional label generated as in Algorithm 3. This additional label has bottom of stack bit (denoted by S in Figure 2) set. These labels are stacked immediately onto the packet and the path labels for routing the packets to appropriate intermediary PEs are added. Figure 3 also shows these path labels used by the data packet to reach PEne. When the packet passes through the core of an intermediary AS involved in model "C", or through the network connecting the intermediary AS, the

intruder or the attacker has the capability to inspect the labels and the payload. However, the proposed scheme prevents the attacker from guessing the right combination of the labels. We can increase the size of the additional inner-labels thereby reducing threats from polynomial time attacks.

2.3 SIMULATION AND IMPLEMENTATION

In this section, we present the preliminary simulation results on performance, comparing the label-hopping technique with deep packet inspection where we encrypt and decrypt the complete packet. We also briefly highlight some implementation issues.

2.3.1 Simulation

Implementing the label-hopping scheme for all set of FECs belonging to any or all VPN service instances may cause throughput degradation. This is because the hashdigest computation and derivation of the inner-label / additional inner label calculation can be computation intensive. We therefore compared our technique by choosing a part of the payload as input to our hashing algorithm. We simulated our algorithm on a 2.5 GHz processor Intel dual processor quad core machine. We compared the performance of the label-hopping technique with a deep packet inspection technique where the complete packet was encrypted before transmission and decrypted on reception. These simulation figures indicate that we were able to process 10 million packets per second when we used 64-byte for hashing on a payload of size 1024 bytes. For a hash using 128-byte, we were able to process about 6.3 million packets per second. However with a deep packet inspection where we encrypted and decrypted the complete packet, we were able to process only about 1 million packets per second. In cases where performance becomes a bottleneck, this label-hopping scheme can be applied to specific traffic which are mission-critical, sensitive and most likely need to be protected as they travel from the PEfa to the PEn. Selective application of this service which could be offered as a premium for a selected set of FECs is a suitable option, thereby protecting the traffic of organizations that are paranoid about the integrity of the switched traffic into their VPN sites.

2.3.2 Implementation

We are modifying the open source Quagga router software on Linux to implement our scheme. One of the concerns in the scheme is the use of payload for generating the random inner label / additional label. If the payload does not vary between two packets then the control-plane exchanges have to be renegotiated with a different algorithm to be used for the hashing for the subsequent packets. The other concern in

the scheme is to tackle the problem of fragmentation that can occur along the path from PEfa to PEn. We can fragment the packet at PEfa and ensure that the size of the packet is fixed before transmission. We could also employ the Path Maximum Transfer Unit (Path-MTU) discovery process so that packets do not get split into multiple fragments. If packets are fragmented this scheme fails. However, networks usually employ the Path-MTU discovery process to prevent fragmentation and hence this problem may not occur.

2.3.3 Running the PTP LSP and label hopping at the ASBRs

The ASes participating in the inter-AS L3VPN Option-C type service connect with each other using ASBRs that connect one AS to another.

It would be prudent to run the PTP LSP and the label-hopping algorithm between the ASBRs instead of between the PEs. Since these ASBRs are usually one-hop away from each other or in the worst case a couple of hops away, the granularity of the time slices can be a lot more finer than when running between the PEs. At more granular time slices it will be even harder for an attacker to pump in packets that utilize the slack of + or - microseconds or milliseconds configured in Algorithm 4.

Hence spoofing and replay attacks are less likely to succeed.

To make it clear the innermost label which is the hash digest computed on the first 128 or 64 byte portion of the payload which is binary anded with an arbitrary bit pattern known to both PEs in the topology, serves as an added binary pattern which has to be guessed by the intruder intending to spoof the packet into the VPN's PE onto the CE.

Thus the effective label space that has to be guessed by the intruder is the label for that time slice and the binary pattern computed on the payload (result of the hash-digest ANDed with the arbitrary bit pattern).

This makes it essentially a 40 bit label space. The hash-digest was not intended to be a ICV. It could serve as an ICV as well.

Since the binary pattern exchanged through the control plane is not known to the intruder, and the hash algorithm used is not known to the intruder (unless of course both of them are compromised in the control plane exchange which is of course secure) the resulting innermost label extends the label space to 40 bits (including the label for that time slice) that has to be guessed.

As to whether there might be a flood of replay packets with the + or

- 1 time slice being in place, the previous label used would be known but the one after the current time slice would be hard to guess owing to the random number generation function being used to determine which the next label should be. It should be possible to jam for that time slice with the same packet with the 2 labels (previous and current) for that time slice being repeated again and again. This is solved by finely granularizing the time slices to microseconds or milliseconds. This is especially the case if the scheme is run between the ASBRs and not between the PEs.

2.4 CONCLUSION AND FUTURE WORK

In this paper, we proposed a label-hopping scheme for inter-provider BGP-based MPLS VPNs that employ MPE-BGP multi-hop control-plane exchanges. In such an environment, without label-hopping, the data-plane is subject to spoofing attacks.

The technique proposed uses a time-based label hopping scheme in addition to the use of the payload to generate an inner label to prevent attackers from easily deciphering labels and their respective VPNs. The scheme is less computationally intensive than encryption-based methods. It prevents the spoofed packets from getting into a VPN site even if the attacker is in the core or at an intervening link between ISPs. In our scheme, we chose the time instant that the packet leaves the first Provider Edge on the far end and this time instant serves as the variable component that the attacker cannot decipher. This requires the use of time synchronization mechanism. This is provided by the PTP LSP constructed for this purpose.

2.5 ACKNOWLEDGEMENTS

The authors would like to acknowledge the UK EP-SRC Digital Economy Programme and the Government of India Department of Science and Technology (DST) for funding given to the IU-ATC.

3 Security Considerations

The main objective of this proposal is to secure the Inter-Provider MPLS VPN Model-C data plane by preventing spoofing attacks and other unidirectional attacks against the customer site in this model. The suggestions and algorithms provided will mitigate these attacks to a large extent. The attacker will have many barriers to break through before he/she can successfully mount an attack against the customer site in this model with these algorithms implemented. The availability of TicToc as a method of clocking helps a great deal in this direction.

4 IANA Considerations

Appropriate IANA indicators would have to be provided to exchange the set of values that Algorithm 1 outlines in order to implement this scheme.

5 References

5.1 Normative References

5.2 Informative References

- [1] S. Alouneh, A. En-Nouaary and A. Agarwal, "MPLS security: an approach for unicast and multicast environments", *Annals of Telecommunications*, Springer, vol. 64, no. 5, June 2009, pp. 391-400, doi:10.1007/s12243-009-0089-y.
- [2] M. H. Behringer and M. J. Morrow, "MPLS VPN security", Cisco Press, June 2005, ISBN-10: 1587051834.
- [3] B. Daugherty and C. Metz, "Multiprotocol Label Switching and IP, Part 1, MPLS VPNS over IP Tunnels", *IEEE Internet Computing*, May-June 2005, pp. 68-72, doi: 10.1109/MIC.2005.61.
- [4] L. Fang, N. Bitar, J. L. Le Roux and J. Miles, "Interprovider IP-MPLS services: requirements, implementations, and challenges", *IEEE Communications Magazine*, vol. 43, no. 6, June 2005, pp. 119-128, doi: 10.1109/MCOM.2005.1452840.
- [5] C. Lin and W. Guowei, "Security research of VPN

technology based on MPLS", Proceedings of the Third International Symposium on Computer Science and Computational Technology (ISCSCT 10), August 2010, pp. 168-170, ISBN- 13:9789525726107.

[6] Y. Rekhter, B. Davie, E. Rosen, G. Swallow, D. Farinacci and D. Katz, "Tag switching architecture overview", Proceedings of the IEEE, vol. 85, no. 12, December 1997, pp. 1973-1983, doi:10.1109/5.650179.

[7] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, Standard Track, February, 2006.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to algorithms", 3rd edition, MIT Press, September 2009, ISBN-10:0262033844.

[9] C. Semeria, "RFC 2547bis: BGP/MPLS VPN fundamentals", Juniper Networks white paper, March 2001.

[10] Advance MPLS VPN Security Tutorials [Online], Available:
"http://etutorials.org/Networking/MPLS+VPN+security/Part+II+Advanced+MPLS+VPN+Security+Issues/", [Accessed: 10th December 2011]

[11] Inter-provider MPLS VPN models [Online], Available:
"http://mpls-configuration-on-cisco-iossoftware.org.ua/1587051990/ ch07lev1sec4.html", [Accessed 10th December 2011]

[12] Davari.S et.al, Transporting PTP messages (1588) over MPLS networks, "http://datatracker.ietf.org/doc/draft-ietf-tictoc-1588overmpls/?include_text=1", Work in Progress, October 2011.

Authors' Addresses

Shankar Raman
Department of Computer Science and Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: mjsraman@cse.iitm.ac.in

Balaji Venkat Venkataswami
Department of Electrical Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: balajivenkat299@gmail.com

Prof.Gaurav Raina
Department of Electrical Engineering
IIT Madras
Chennai - 600036
TamilNadu
India

EMail: gaurav@ee.iitm.ac.in

Network Working Group
Internet Draft
Intended status: Experimental
Expires: August 2014

A. Shpiner
Technion - Israel Institute of Technology
R. Tse
C. Schelp
PMC-Sierra
T. Mizrahi
Marvell
February 11, 2014

Multi-Path Time Synchronization
draft-shpiner-multi-path-synchronization-03.txt

Abstract

Clock synchronization protocols are very widely used in IP-based networks. The Network Time Protocol (NTP) has been commonly deployed for many years, and the last few years have seen an increasingly rapid deployment of the Precision Time Protocol (PTP). As time-sensitive applications evolve, clock accuracy requirements are becoming increasingly stringent, requiring the time synchronization protocols to provide high accuracy. Slave Diversity is a recently introduced approach, where the master and slave clocks (also known as server and client) are connected through multiple network paths, and the slave combines the information received through all paths to obtain a higher clock accuracy compared to the conventional one-path approach. This document describes a multi-path approach to PTP and NTP over IP networks, allowing the protocols to run concurrently over multiple communication paths between the master and slave clocks. The multi-path approach can significantly contribute to clock accuracy, security and fault protection. The Multi-Path Precision Time Protocol (MPPTP) and Multi-Path Network Time Protocol (MPNTP) define an additional layer that extends the existing PTP and NTP without the need to modify these protocols. MPPTP and MPNTP also allow backward compatibility with nodes that do not support the multi-path extension.

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/lid-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 August 11, 2014.

Copyright Notice

Copyright (c) 2014 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	3
2. Conventions Used in this Document	5
2.1. Abbreviations	5
2.2. Terminology	5
3. Multiple Paths in IP Networks	5
3.1. Load Balancing	5
3.2. Using Multiple Paths Concurrently	5
3.3. Two-Way Paths	6
4. Solution Overview	6
4.1. Path Configuration and Identification	6
4.2. Combining	7
5. Multi-Path Time Synchronization Protocols over IP Networks ...	7
5.1. Single-Ended Multi-Path Synchronization	8
5.1.1. Single-Ended MPPTP Synchronization Message Exchange	8
5.1.2. Single-Ended MPNTP Synchronization Message Exchange	9
5.2. Dual-Ended Multi-Path Synchronization	10
5.2.1. Dual-Ended MPPTP Synchronization Message Exchange .	10

5.2.2. Dual-Ended MPNTP Synchronization Message Exchange .	11
5.3. Using Traceroute for Path Discovery	12
5.4. Using Unicast Discovery for MPPTP	12
6. Combining Algorithm	13
6.1. Averaging	13
6.2. Switching / Dynamic Algorithm	13
6.3. NTP-like Filtering-Clustering-Combining Algorithm	13
7. Security Considerations	14
8. IANA Considerations	14
9. Acknowledgments	14
10. References	14
10.1. Normative References	14
10.2. Informative References	15

1. Introduction

The two most common time synchronization protocols in IP networks are the Network Time Protocol [NTP], and the Precision Time Protocol (PTP), defined in the IEEE 1588 standard [IEEE1588].

The accuracy of the time synchronization protocols directly depends on the stability and the symmetry of propagation delays on both directions between the master and slave clocks. Depending on the nature of the underlying network, time synchronization protocol packets can be subject to variable network latency or path asymmetry (e.g. [ASYMMETRY], [ASYMMETRY2]). As time sensitive applications evolve, accuracy requirements are becoming increasingly stringent.

Using a single network path in a clock synchronization protocol closely ties the slave clock accuracy to the behavior of the specific path, which may suffer from temporal congestion, faults or malicious attacks. Relying on multiple clock servers as in NTP solves these problems, but requires active maintenance of multiple accurate sources in the network, which is not always possible. The usage of Transparent Clocks (TC) in PTP solves the congestion problem by eliminating the queueing time from the delay calculations, but requires the intermediate routers and switches to support the TC functionality, which is not always the case.

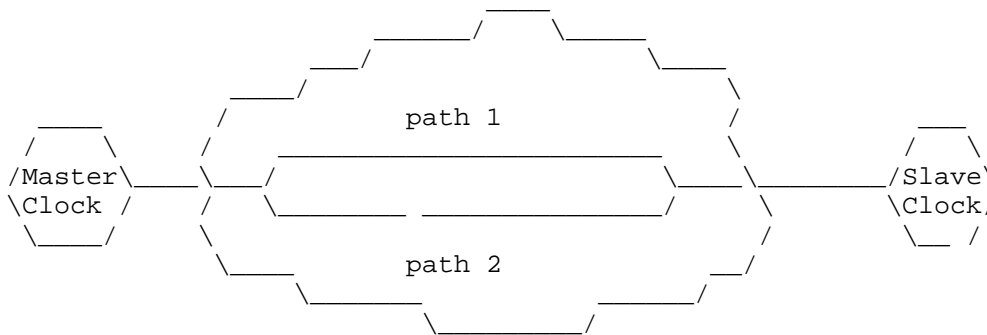


Figure 1 Multi-Path Connection

Since master and slave clocks are often connected through more than one path in the network, as shown in Figure 1, [SLAVEDIV] suggested that a time synchronization protocol can be run over multiple paths, providing several advantages. First, it can significantly increase the clock accuracy as shown in [SLAVEDIV]. Second, this approach provides additional security, allowing mitigating man-in-the-middle attacks against the time synchronization protocol [DELAY-ATT]. Third, using multiple paths concurrently provides an inherent failure protection mechanism.

This document introduces Multi-Path PTP (MPPTP) and Multi-Path NTP (MPNTP), respectively. These extensions are defined at the network layer and do not require any changes in the PTP or in the NTP protocols.

MPPTP and MPNTP are defined over IP networks. As IP networks typically combine ECMP routing, this property is leveraged for the multiple paths used in MPPTP and MPNTP. The key property of the multi-path extension is that clocks in the network can use more than one IP address. Each {master IP, slave IP} address pair defines a path. Depending on the network topology and configuration, the IP combination pairs can form multiple diverse paths used by the multi-path synchronization protocols.

This document introduces two variants for each of the two multi-path protocols; a variant that requires both master and slave nodes to support the multi-path protocol, referred to as the dual-ended variant, and a backward compatible variant that allows a multi-path

clock to connect to a conventional single-path clock, referred to as the single-ended variant.

2. Conventions Used in this Document

2.1. Abbreviations

ECMP	Equal Cost Multiple Path
LAN	Local Area Network
MPNTP	Multi-Path Network Time Protocol
MPPTP	Multi-Path Precision Time Protocol
NTP	Network Time Protocol
PTP	Precision Time Protocol

2.2. Terminology

In the NTP terminology, a time synchronization protocol is run between a client and a server, while PTP uses the terms master and slave. Throughout this document, the sections that refer to both PTP and NTP generically use the terms master and slave.

3. Multiple Paths in IP Networks

3.1. Load Balancing

Traffic sent across IP networks is often load balanced across multiple paths. The load balancing decisions are typically based on packet header fields: source and destination addresses, Layer 4 ports, the Flow Label field in IPv6, etc. Three common load balancing criteria are per-destination, per-flow and per-packet. The per-destination load balancers take a load balancing decision based on the destination IP address. Per-flow load balancers use various fields in the packet header, e.g., IP addresses and Layer 4 ports, for the load balancing decision. Per-packet load balancers use flow-blind techniques such as round-robin without basing the choice on the packet content.

3.2. Using Multiple Paths Concurrently

To utilize the diverse paths that traverse per-destination load-balancers or per-flow load-balancers, the packet transmitter can vary

the IP addresses in the packet header. The analysis in [PARIS2] shows that a significant majority of the flows on the internet traverse per-destination or per-flow load-balancing. It presents statistics that 72% of the flows traverse per-destination load balancing and 39% of the flows traverse per-flow load-balancing, while only a negligible part of the flows traverse per-packet load balancing. These statistics show that the vast majority of the traffic on the internet is load balanced based on packet header fields.

The approaches in this draft are based on varying the source and destination IP addresses in the packet header. Possible extensions have been considered that also vary the UDP ports. However some of the existing implementations of PTP and NTP use fixed UDP port values in both the source and destination UDP port fields, and thus do not allow this approach.

3.3. Two-Way Paths

A key property of IP networks is that packets forwarded from A to B do not necessarily traverse the same path as packets from B to A. Thus, we define a two-way path for a master-slave connection as a pair of one-way paths: the first from master to slave and the second from slave to master.

If possible, a traffic engineering approach can be used to verify that time synchronization traffic is always forwarded through bidirectional two-way paths, i.e., that each two-way path uses the same route on the forward and reverse directions, thus allowing propagation time symmetry. However, in the general case two-way paths do not necessarily use the same path for the forward and reverse directions.

4. Solution Overview

The multi-path time synchronization protocols we present are comprised of two building blocks; one is the path configuration and identification, and the other is the algorithm used by the slave to combine the information received from the various paths.

4.1. Path Configuration and Identification

The master and slave clocks must be able to determine the path of transmitted protocol packets, and to identify the path of incoming protocol packets. A path is determined by a {master IP, slave IP} address pair. The synchronization protocol message exchange is run independently through each path.

Each IP address pair defines a two-way path, and thus allows the clocks to bind a transmitted packet to a specific path, or to identify the path of an incoming packet.

If possible, the routing tables across the network should be configured with multiple traffic engineered paths between the pair of clocks. By carefully configuring the routers in such networks it is possible to create diverse paths for each of the IP address pairs between two clocks in the network. However, in public and provider networks the load balancing behavior is hidden from the end users. In this case the actual number of paths may be less than the number of IP address pairs, since some of the address pairs may share common paths.

4.2. Combining

Various methods can be used for combining the time information received from the different paths. This document surveys several combining methods in Section 5.4. The output of the combining algorithm is the accurate time offset.

5. Multi-Path Time Synchronization Protocols over IP Networks

This section presents two variants of MPPTP and MPNTP; single-ended multi-path time synchronization and dual-ended multi-path time synchronization. In the first variant, the multi-path protocol is run only by the slave and the master is not aware of its usage. In the second variant, all clocks must support the multi-path protocol.

The dual-ended protocol provides higher path diversity by using multiple IP addresses at both ends, the master and slave, while the single-ended protocol only uses multiple addresses at the slave. On the other hand, the dual-ended protocol can only be deployed when both the master and the slave support this protocol. Dual-ended and single-ended protocols can co-exist in the same network. Each slave selects the connection(s) it wants to make with the available masters. A dual-ended slave could switch to single-ended mode if it does not see any dual-ended masters available. A single-ended slave could connect to a single IP address of a dual-ended master.

Multi-path time synchronization, in both variants, requires clocks to use multiple IP addresses. If possible, the set of IP addresses for each clock should be chosen in a way that enables the establishment of paths that are the most different. It is applicable if the load balancing rules in the network are known. Using multiple IP addresses introduces a tradeoff. A large number of IP addresses allows a large number of diverse paths, providing the advantages of slave diversity

discussed in Section 1. On the other hand, a large number of IP addresses is more costly, requires the network topology to be more redundant, and exacts extra management overhead.

The descriptions in this section refer to the end-to-end scheme of PTP, but are similarly applicable to the peer-to-peer scheme. The MPNTP protocol described in this document refers to the NTP client-server mode, although the concepts described here can be extended to include the symmetric variant as well.

Multi-path synchronization protocols by nature require protocol messages to be sent as unicast. Specifically in PTP, the following messages must be sent as unicast in MPPTP: Sync, Delay_Req, Delay_Resp, PDelay_Req, PDelay_Resp, Follow_Up, and PDelay_Resp_Follow_Up. Note that [IEEE1588] allows these messages to be sent either as multicast or as unicast.

5.1. Single-Ended Multi-Path Synchronization

In the single-ended approach, only the slave is aware of the fact that multiple paths are used, while the master is agnostic to the usage of multiple paths. This approach allows a hybrid network, where some of the clocks are multi-path clocks, and others are conventional one-path clocks. A single-ended multi-path clock presents itself to the network as N independent clocks, using N IP addresses, as well as N clock identity values (in PTP). Thus, the usage of multiple slave identities by a slave clock is transparent from the master's point of view, such that it treats each of the identities as a separate slave clock.

5.1.1. Single-Ended MPPTP Synchronization Message Exchange

The single-ended MPPTP message exchange procedure is as follows.

- o Each single-ended MPPTP clock has a fixed set of N IP addresses and N corresponding clockIdentities. Each clock arbitrarily defines one of its IP addresses and clockIdentity values as the clock primary identity.
- o A single-ended MPPTP port sends Announce messages only from its primary identity, according to the BMC algorithm.
- o The BMC algorithm at each clock determines the master, based on the received Announce messages.

- o A single-ended MPPTP port that is in the 'slave' state uses unicast negotiation to request the master to transmit unicast messages to each of the N slave clock identities. The slave port periodically sends N Signaling messages to the master, using each of its N identities. The Signaling message includes the REQUEST_UNICAST_TRANSMISSION_TLV.
- o The master periodically sends unicast Sync messages from its primary identity, identified by the sourcePortIdentity and IP address, to each of the slave identities.
- o The slave, upon receiving a Sync message, identifies its path according to the destination IP address. The slave sends a Delay_Req unicast message to the primary identity of the master. The Delay_Req is sent using the slave identity corresponding to the path the Sync was received through. Note that the rate of Delay_Req messages may be lower than the Sync message rate, and thus a Sync message is not necessarily followed by a Delay_Req.
- o The master, in response to a Delay_Req message from the slave, responds with a Delay_Resp message using the IP address and sourcePortIdentity from the Delay_Req message.
- o Upon receiving the Delay_Resp message, the slave identifies the path using the destination IP address and the requestingPortIdentity. The slave can then compute the corresponding path delay and the offset from the master.
- o The slave combines the information from all negotiated paths.

5.1.2. Single-Ended MPNTP Synchronization Message Exchange

The single-ended MPNTP message exchange procedure is as follows.

- o A single-ended MPNTP client has N separate identities, i.e., N IP addresses. The assumption is that the server information, including its IP address is known to the NTP clients.
- o A single-ended MPNTP client initiates the NTP protocol with an NTP server N times, using each of its N identities.
- o The NTP protocol is maintained between the server and each of the N client identities.
- o The client sends NTP messages to the master using each of its N identities.

- o The server responds to the client's NTP messages using the IP address from the received NTP packet.
- o The client, upon receiving an NTP packet, uses the IP destination address to identify the path it came through, and uses the time information accordingly.
- o The client combines the information from all paths.

5.2. Dual-Ended Multi-Path Synchronization

In dual-ended multi-path synchronization each clock has N IP addresses. Time synchronization messages are exchanged between some of the combinations of {master IP, slave IP} addresses, allowing multiple paths between the master and slave. Note that the actual number of paths between the master and slave may be less than the number of chosen {master, slave} IP address pairs.

Once the multiple two-way connections are established, a separate synchronization protocol exchange instance is run through each of them.

5.2.1. Dual-Ended MPPTP Synchronization Message Exchange

The dual-ended MPPTP message exchange procedure is as follows.

- o Every clock has N IP addresses, but uses a single clockIdentity.
- o The BMC algorithm at each clock determines the master. The master is identified by its clockIdentity, allowing other clocks to know the multiple IP addresses it uses.
- o When a clock sends an Announce message, it sends it from each of its IP addresses with its clockIdentity.
- o A dual-ended MPPTP port that is in the 'slave' state uses unicast negotiation to request the master to transmit unicast messages to some or all of its N_s IP addresses. This negotiation is done individually between a slave IP address and the corresponding master IP address that the slave desires a connection with. The slave port periodically sends Signaling messages to the master, using some or all of its N_s IP addresses as source, to the corresponding master's N_m IP addresses. The Signaling message includes the REQUEST_UNICAST_TRANSMISSION_TLV.

- o The master periodically sends unicast Sync messages from each of its IP addresses to the corresponding slave IP addresses for which a unicast connection was negotiated.
- o The slave, upon receiving a Sync message, identifies its path according to the {source, destination} IP addresses. The slave sends a Delay_Req unicast message, swapping the source and destination IP addresses from the Sync message. Note that the rate of Delay_Req messages may be lower than the Sync message rate, and thus a Sync message is not necessarily followed by a Delay_Req.
- o The master, in response to a Delay_Req message from the slave, responds with a Delay_Resp message using the sourcePortIdentity from the Delay_Req message, and swapping the IP addresses from the Delay_Req.
- o Upon receiving the Delay_Resp message, the slave identifies the path using the {source, destination} IP address pair. The slave can then compute the corresponding path delay and the offset from the master.
- o The slave combines the information from all negotiated paths.

5.2.2. Dual-Ended MPNTP Synchronization Message Exchange

The MPNTP message exchange procedure is as follows.

- o Each NTP clock has a set of N IP addresses. The assumption is that the server information, including its multiple IP addresses is known to the NTP clients.
- o The MPNTP client chooses N_{svr} of the N server IP addresses and N_{c} of the N client IP addresses and initiates the $N_{\text{svr}} \times N_{\text{c}}$ instances of the protocol, one for each {server IP, client IP} pair, allowing the client to combine the information from the $N_{\text{s}} \times N_{\text{c}}$ paths.
(N_{svr} and N_{c} indicate the number of IP addresses of the server and client, respectively, which a client chooses to connect with)
- o The client sends NTP messages to the master using each of the source-destination address combinations.
- o The server responds to the client's NTP messages using the IP address combination from the received NTP packet.

- o Using the {source, destination} IP address pair in the received packets, the client identifies the path, and performs its computations for each of the paths accordingly.
- o The client combines the information from all paths.

5.3. Using Traceroute for Path Discovery

The protocols presented above use multiple IP addresses in a single clock to create multiple paths. However, although each two-way path is defined by a different {master, slave} address pair, some of the IP address pairs may traverse exactly the same network path, making them redundant. Traceroute-based path discovery can be used for filtering only the IP addresses that obtain diverse paths. 'Paris Traceroute' [PARIS] and 'TraceFlow' [TRACEFLOW] are examples of tools that discover the paths between two points in the network.

The Traceroute-based filtering can be implemented by both master and slave nodes, or it can be restricted to run only on slave nodes to reduce the overhead on the master. For networks that guarantee the path of the timing packets in the forward and reverse direction are the same, path discovery should only be performed at the slave.

5.4. Using Unicast Discovery for MPPTP

As presented above, MPPTP uses Announce messages and the BMC algorithm to discover the master. The unicast discovery option of PTP can be used as an alternative.

When using unicast discovery the MPPTP slave ports maintain a list of the IP addresses of the master. The slave port uses unicast negotiation to request unicast service from the master, as follows:

- o In single-ended MPPTP, the slave uses unicast negotiation from each of its identities to the master's (only) identity.
- o In dual-ended MPPTP, the slave uses unicast negotiation from its IP addresses, each to a corresponding master IP address to request unicast synchronization messages.

Afterwards, the message exchange continues as described in sections 5.1.1. and 5.2.1.

The unicast discovery option can be used in networks that do not support multicast or in networks in which the master clocks are known in advance. In particular, unicast discovery avoids multicasting Announce messages.

6. Combining Algorithm

Previous sections discussed the methods of creating the multiple paths and obtaining the time information required by the slave algorithm. This section discusses the algorithm used to combine this information into a single accurate time estimate. Note that the choice of the combining algorithm is local to the slave, and does not affect the interoperability of the protocol. Several combining methods are examined next.

6.1. Averaging

In the first method the slave performs an autonomous time computation for each of the master-slave paths, and obtains the combined time by simply averaging the separate instances. This method can be further enhanced by adding weights to each of the paths. For example, a reasonable weighting choice is to use an inverse of the round-trip delay between the peers. Another option is to use the inverse of the path delay variance, which is approximately the maximum likelihood estimator under certain assumptions [WEIGHT-MEAN].

6.2. Switching / Dynamic Algorithm

The switching and dynamic algorithms are presented in [SLAVEDIV]. The switching algorithm periodically chooses a primary path, and performs all time computations based on the protocol packets received through the primary path. The primary path is defined as the path with the minimal distance between the sampled delay and the average delay. The dynamic algorithm dynamically chooses between the result of the switching algorithm and the averaging.

6.3. NTP-like Filtering-Clustering-Combining Algorithm

NTP ([NTP], [NTP2]) provides an efficient algorithm of combining offset samples from multiple peers. The same approach can be used in MPPTP and MPNTP.

In the MPNTP, the selection and combining algorithms treat the offset samples from multiple paths as NTP treats samples from distinct peers. The rest of the selection and combining algorithms, as well as clock control logic is the same as in conventional NTP. In MPPTP, a similar approach to NTP can be adopted.

The combining algorithm [NTP3] contains three steps: filtering, selection and clustering.

In the filtering step, the best of the last n (usually $n=8$) samples of each peer is chosen. The choice criterion is the combination of a round trip delay estimate of the sample and the distance from the average offset of all n samples of a peer.

In the selection step the peers are divided into two groups: true-chimers and false tickers.

The clustering step chooses a subset of the true-chimers, whose peer jitter (the variance of peer offset samples) is smaller than the total select jitter of all selected peer offsets (the variance of the best offset of the selected peers).

The offset samples that passed through the three steps are combined by a weighted average into a single offset estimate. Detailed explanations are provided in [NTP2],[NTP3].

7. Security Considerations

The security aspects of time synchronization protocols are discussed in detail in [TICTOCSEC]. The methods describe in this document propose to run a time synchronization protocol through redundant paths, and thus allow to detect and mitigate man-in-the-middle attacks, as described in [DELAY-ATT].

8. IANA Considerations

There are no IANA actions required by this document.

RFC Editor: please delete this section before publication.

9. Acknowledgments

The authors gratefully acknowledge the useful comments provided by Peter Meyer and Doug Arnold, as well as other comments received from the TICTOC working group participants.

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

10. References

10.1. Normative References

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

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

10.2. Informative References

- [ASSYMETRY] Yihua He and Michalis Faloutsos and Srikanth Krishnamurthy and Bradley Huffaker, "On routing asymmetry in the internet", IEEE Globecom, 2005.
- [ASSYMETRY2] Abhinav Pathak, Himabindu Pucha, Ying Zhang, Y. Charlie Hu, and Z. Morley Mao, "A measurement study of internet delay asymmetry", PAM'08, 2008.
- [DELAY-ATT] T. Mizrahi, "A Game Theoretic Analysis of Delay Attacks against Time Synchronization Protocols", ISPCS, 2012.
- [NTP2] Mills, D.L., "Internet time synchronization: the Network Time Protocol", IEEE Trans. Communications COM-39, 10 (October 1991), 1482-1493.
- [NTP3] Mills, D.L., "Improved algorithms for synchronizing computer network clocks", IEEE/ACM Trans. Networks 3, 3(June 1995), 245-254.
- [PARIS] Brice Augustin, Timur Friedman and Renata Teixeira, "Measuring Load-balanced Paths in the Internet", IMC, 2007.
- [PARIS2] B. Augustin, T. Friedman, and R. Teixeira, "Measuring Multipath Routing in the Internet", IEEE/ACM Transactions on Networking, 19(3), p. 830 - 840, June 2011.
- [SLAVEDIV] T. Mizrahi, "Slave Diversity: Using Multiple Paths to Improve the Accuracy of Clock Synchronization Protocols", ISPCS, 2012.
- [TICTOCSEC] T. Mizrahi, K. O'Donoghue, "Security Requirements of Time Protocols in Packet Switched Networks", IETF, draft-ietf-tictoc-security-requirements, work in progress, 2013.
- [TRACEFLOW] J. Narasimhan, B. V. Venkataswami, R. Groves and P. Hoose, "Traceflow", IETF, draft-janapath-intarea-traceflow, work in progress, 2012.

[WEIGHT-MEAN] http://en.wikipedia.org/wiki/Weighted_mean#Dealing_with_variance

Authors' Addresses

Alex Shpiner
Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa, 32000 Israel

Email: shalex@tx.technion.ac.il

Richard Tse
PMC-Sierra
8555 Baxter Place
Burnaby, BC
Canada
V5A 4V7

Email: Richard.Tse@pmcs.com

Craig Schelp
PMC-Sierra
8555 Baxter Place
Burnaby, BC
Canada
V5A 4V7

Email: craig.schelp@pmcs.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam, 20692 Israel

Email: talmi@marvell.com

NTP Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 25, 2013

D. Sibold
PTB
S. Roettger
TU-BS
February 23, 2013

Network Time Protocol: autokey Version 2 Specification
draft-sibold-autokey-02

Abstract

This document describes a security protocol that enables authenticated time synchronization using Network Time Protocol (NTP). Autokey Version 2 obsoletes NTP autokey protocol RFC 5906 [RFC5906] which suffers from various security vulnerabilities. Its design considers the special requirements that are related to the task of precise timekeeping.

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 August 25, 2013.

Copyright Notice

Copyright (c) 2013 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. Differences from the original autokey	3
2. Security Threats	3
3. Objectives	3
4. Terms and abbreviations	4
5. Autokey Overview	4
5.1. Symmetric and Client/Server Mode	4
5.2. Broadcast Mode	4
6. Protocol Sequence	5
6.1. Association Message	5
6.2. Certificate Message	5
6.3. Cookie Message	6
6.4. Broadcast Parameter Message	6
6.5. Time Request Message	6
6.6. Broadcast Message	6
7. Hash algorithms and MAC generation	7
7.1. Hash algorithms	7
7.2. MAC Calculation	7
8. Server Seed Considerations	8
8.1. Server Seed algorithm	8
8.2. Server Seed Live Time	8
9. IANA Considerations	8
10. Security Considerations	8
11. Acknowledgements	8
12. References	8
12.1. Normative References	8
12.2. Informative References	9
Appendix A. TICTOC Security Requirements	9
Appendix B. Broadcast Mode	10
Authors' Addresses	10

1. Introduction

In NTP [RFC5905] the autokey protocol [RFC5906] was introduced to provide authenticity to NTP servers and to ensure integrity of time synchronization. It is designed to meet the specific communication requirements of precise timekeeping and therefore does not compromise timekeeping precision.

This document focuses on a new definition of the autokey protocol for NTP, autokey version 2. The necessity to renew the autokey specification arises from various severe security vulnerabilities that have been found in a thorough analysis of the protocol [Roettger]. The new specification is based on the same assumptions as the original autokey specification. In particular, the prerequisite is that precise timekeeping can only be accomplished with stateless time synchronization communication, which excludes standard security protocols like IPSec or TLS. This prerequisite corresponds with the requirement that a security mechanism for timekeeping must be designed in such a way that it does not degrade the quality of the time transfer [I-D.ietf-tictoc-security-requirements].

1.1. Differences from the original autokey

Autokey version 2 is a major redraft of the original autokey specification. It is intended to mitigate security vulnerabilities of the original specification and it is based on the suggestions in the analysis of Roettger [Roettger]. The major changes are:

- o The bit length of server seed and cookie has been increased.
- o The IP addresses of the synchronization partners in the calculation of the cookie have been replaced by the hash value of the client's public key.
- o The identity schemes for the verification of the NTP server authenticity have been replaced by a hierarchical public key infrastructure (PKI) based on X.509 certificates.

2. Security Threats

A profound analysis of security threats and requirements for NTP and Precision Time Protocol (PTP) can be found in the I-D [I-D.ietf-tictoc-security-requirements].

3. Objectives

The objectives of the autokey specifications are as follows:

- o Authenticity: Autokey enables the client to authenticate its NTP server or peer.
- o Integrity: Autokey protects the integrity of time synchronization packets via a message authentication code (MAC).
- o Confidentiality: Autokey does not provide confidentiality

protection of the NTP packets.

- o Modes of operation: All operational modes of NTP are supported (client server, symmetric, broadcast).
- o Hybrid mode: Both secure and insecure communication modes are possible for NTP servers and clients, respectively.
- o Compatibility:
 - * Interoperation with autokey version 1 is not given.
 - * NTP associations without authentication shall not be affected.
 - * An NTP server that does not support autokey version 2 shall not be affected by autokey version 2 authentication requests.

4. Terms and abbreviations

- o Throughout this document the term "autokey" refers to autokey version 2.
- o TESLA: Time efficient stream loss-tolerant authentication

5. Autokey Overview

5.1. Symmetric and Client/Server Mode

Authenticity and integrity of the NTP packets are ensured by a Message Authentication Code (MAC), which is attached to the NTP packet. The calculation of the MAC includes the whole NTP packet and the cookie which is shared between client and server. It is calculated according to:

$$\text{cookie} = \text{MSB}_{128} (\text{H}(\text{server seed} || \text{H}(\text{public key of client}))),$$

where `||` indicates concatenation and in which `H` is a hash algorithm. The function `MSB128` cuts off the 128 most significant bits of the result of the hash function. The server seed is a 128 bit random value of the server, which has to be kept secret. The cookie thus never changes. The server seed has to be refreshed periodically. The server does not keep a state of the client. Therefore it has to recalculate the cookie each time it receives a request from the client. To this end, the client has to attach the hash value of its public key to each request (see Section 6.5).

5.2. Broadcast Mode

Just as in the case of the client server mode and symmetric mode, authenticity and integrity of the NTP packets are ensured by a MAC, which is attached to the NTP packet by the sender. The verification of the authenticity is based on the TESLA protocol [RFC4082]. TESLA is based on a one-way chain of keys, where each key is the output of a one-way function applied on the previous key in the chain. The last element of the chain is shared securely with all clients. The server splits time into intervals of uniform duration and assigns each key to an interval in reverse order, starting with the penultimate. At each time interval, the server sends an NTP broadcast packet appended by a MAC, calculated using the corresponding key, and the key of the previous interval. The client verifies the MAC by buffering the packet until the disclosure of the key in the next interval. In order to be able to verify the validity of the key, the client has to be loosely time synchronized to the server. This has to be accomplished during the initial client server exchange between broadcast client and server.

6. Protocol Sequence

6.1. Association Message

The protocol sequence starts with the association message, in which the client sends an NTP packet with an extension field of type association. It contains the hostname of the client and a status word which contains the algorithms used for the signatures and the status of the connection. The response contains the hostname of the server and the algorithms for the signatures. The server notifies the cryptographic hash algorithms which it supports.

6.2. Certificate Message

In this step, the client receives the certification chain up to the trusted authority (TA). To this end, the client requests the certificate for the subject name (hostname) of the NTP server. The response contains the certificate with the issuer name. If the issuer name is different from the subject name, the client requests the certificate for the issuer. This continues until it receives a certificate which is issued by a TA. The client recognizes the TA because it has a list of certificates which are accepted as TAs. The client has to check that each issuer is authorized to issue new certificates. To this end, the certificates have to include the X.509v3 extension field "CA:TRUE". With the established certification chain the client is able to verify the server signatures and, hence, the authenticity of the server messages with extension fields is ensured.

Discussion:

Note that in this step the client validate the authenticity of its NTP-server only. It does not recursively validate the authenticity of each NTP server on the time synchronization chain. But each NTP server on the time synchronization chain validates the NTP server to which it is synchronized. This conforms to the recursive authentication requirement in the TICTOC security requirements [I-D.ietf-tictoc-security-requirements].

6.3. Cookie Message

The client requests a cookie from the server. It selects a hash algorithm from the list of algorithms supported by the server. The request includes its public key and the selected hash algorithm. The hash of the public key is used by the server to calculate the cookie (see Section 5.1). The response of the server contains the cookie encrypted with the public key.

6.4. Broadcast Parameter Message

In the broadcast mode the client requests the following information from the server:

- o the last key of the one-way key chain,
- o the disclosure schedule of the following keys. This contains:
 - * time interval duration, time at which the next time interval will start and its associated index,
 - * key disclosure delay (number of time intervals for which a key is valid).

The server will sign all transmitted properties so that the client is able to verify their authenticity. For this packet exchange a new extension field "broadcast parameters" is used. The client synchronizes its time with the server in the client server mode and saves an upper bound of its time offset with respect to the time of the server. See Appendix B for more details.

6.5. Time Request Message

The client request includes a new extension field "time request" which contains the hash of its public key. The server needs the hash of the public key to recalculate the cookie for the client. The response is a normal NTP packet without extension field. It contains a MAC.

6.6. Broadcast Message

The NTP broadcast packet includes a new extension field "broadcast message" which contains the disclosed key of the previous disclosure interval (current time interval minus disclosure delay). The NTP packet is appended by a MAC, calculated with the key for the current time interval. When a client receives a broadcast message it has to perform the following tests:

- o Proof that the MAC is based on a key that is not yet disclosed. If verified the packet will be buffered for later authentication otherwise it has to be discarded.
- o The client checks whether it already knows the disclosed key. If not, the client verifies its legitimacy. If falsified the packet has to be discarded.
- o If the disclosed key is legitimate the client verifies the authenticity of any packet that it received during the corresponding time interval. If authenticity of a packet is verified it is released from the buffer. If the verification fails authenticity is no longer given. In this case the client MUST request authentic time from the server by means of a unicast time request message.

See Appendix B or [RFC4082] for a detailed description of the packet verification process.

7. Hash algorithms and MAC generation

7.1. Hash algorithms

Hash algorithms are used at different points: calculation of the cookie and the MAC, and hashing of the public key. The client selects the hash algorithm from the list of hash algorithms which are supported by the server. This list is notified during the association message exchange (Section 6.1). The selected algorithm is used for all hashing processes in the protocol.

In the broadcast mode hash algorithm are used as pseudo random function to construct the one-way key chain.

The list of the server supported hash algorithms has to fulfill following requirements:

- o it MUST NOT contain the MD5 or weaker algorithms,
- o it MUST include SHA-256 or stronger algorithms.

7.2. MAC Calculation

For the calculation of the MAC client and server are using a Keyed-Hash Message Authentication Code (HMAC) approach [RFC2104]. The HMAC is generated with the hash algorithm specified by the client (see Section 7.1).

8. Server Seed Considerations

The server has to calculate a random seed which has to be kept secret and which has to be changed periodically. The server has to generate a seed for each supported hash algorithm.

8.1. Server Seed algorithm

8.2. Server Seed Live Time

9. IANA Considerations

This document makes no request of IANA.

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

10. Security Considerations

The client has to verify the validity of the certificates during the certification message exchange (Section 6.2). Since it generally has no reliable time during this initial communication phase, it is impossible to verify the period of validity of the certificates. Therefore, the client **MUST** use one of the following approaches:

- o The validity of the certificates is preconditioned. Usually this will be the case in corporation networks.
- o The client ensures that the certificates are not revoked. To this end, the client uses the Online Certificate Status Protocol (OCSP) defined in [RFC6277].
- o The client requests a different service to get an initial time stamp in order to be able to verify the certificates' periods of validity. To this end, it can, e.g., use a secure shell connection to a reliable host. Another alternative is to request a time stamp from a Time Stamping Authority (TSA) by means of the Time-Stamp Protocol (TSP) defined in [RFC3161].

11. Acknowledgements

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3161] Adams, C., Cain, P., Pinkas, D. and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", RFC 3161, August 2001.

- [RFC6277] Santesson, S. and P. Hallam-Baker, "Online Certificate Status Protocol Algorithm Agility", RFC 6277, June 2011.

12.2. Informative References

- [I-D.ietf-tictoc-security-requirements]
Mizrahi, T., "Security Requirements of Time Synchronization Protocols in Packet Switched Networks", Internet-Draft draft-ietf-tictoc-security-requirements-04, February 2013.
- [RFC2104] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.
- [RFC4082] Perrig, A., Song, D., Canetti, R., Tygar, J.D. and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction", RFC 4082, June 2005.
- [RFC5905] Mills, D., Martin, J., Burbank, J. and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.
- [RFC5906] Haberman, B. and D. Mills, "Network Time Protocol Version 4: Autokey Specification", RFC 5906, June 2010.
- [Roettger]
Roettger, S., "Analysis of the NTP Autokey Procedures", February 2012.

Appendix A. TICTOC Security Requirements

The following table compares the autokey specifications against the TICTOC security requirements [I-D.ietf-tictoc-security-requirements].

Section	Requirement from I-D tictoc security-requirements-04	Requirement level	Autokey V2
5.1	Clock Identity Authentication and Authorization	MUST	OK
5.1.1	Authentication and Authorization of Masters	MUST	OK
5.1.2	Recursive Authentication and Authorization of Masters (Chain of Trust)	MUST	OK
5.1.3	Authentication and Authorization of Slaves	MAY	-
5.2	Integrity protection.	MUST	OK
5.3	Protection against DoS attacks	SHOULD	-
5.4	Replay protection	MUST	OK (NTP)
5.5.1	Key freshness.	MUST	OK
5.5.2	Security association.	SHOULD	OK
5.5.3	Unicast and multicast associations.	SHOULD	OK
5.6	Performance: no degradation in quality of time transfer.	MUST	OK
	Performance: lightweight computation	SHOULD	OK
	Performance: storage, bandwidth	SHOULD	OK
5.7	Confidentiality protection	MAY	-
5.8	Protection against Packet Delay and Interception Attacks	SHOULD	-
5.9.1	Secure mode	MUST	OK (NTP)
5.9.2	Hybrid mode	MAY	OK (NTP)

Comparison between TICTOC security requirements and autokey.

Appendix B. Broadcast Mode

Authors' Addresses

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

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

Stephen Roettger
Technische Universitaet Braunschweig
Email: stephen.roettger@gmail.com