          UDP Checksum Complement in the Network Time Protocol (NTP)

Abstract

   The Network Time Protocol (NTP) allows clients to synchronize to a
   time server using timestamped protocol messages. To facilitate
   accurate timestamping, some implementations use hardware-based
   timestamping engines that integrate the accurate transmission time
   into every outgoing NTP packet during transmission. Since these
   packets are transported over UDP, the UDP Checksum field is then
   updated to reflect this modification. This document proposes an
   extension field that includes a 2-octet Checksum Complement, allowing
   timestamping engines to reflect the checksum modification in the last
   2 octets of the packet rather than in the UDP Checksum field. The
   behavior defined in this document is interoperable with existing NTP
   implementations.

Status of This Memo

   This document is not an Internet Standards Track specification; it is
   published for examination, experimental implementation, and
   evaluation.

   This document defines an Experimental Protocol for the Internet
   community. This document is a product of the Internet Engineering
   Task Force (IETF). It represents the consensus of the IETF
   community. It has received public review and has been approved for
   publication by the Internet Engineering Steering Group (IESG). Not
   all documents approved by the IESG are a candidate for any level of
   Internet Standard; see Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc7821.

Copyright Notice

Table of Contents

1. Introduction

   The Network Time Protocol [NTPv4] allows clients to synchronize their
   clocks to a time server by exchanging NTP packets. The increasing
   demand for highly accurate clock synchronization motivates
   implementations that provide accurate timestamping.

1.1. Intermediate Entities

   In this document, we use the term "intermediate entity" to refer to
   an entity that resides on the path between the sender and the
   receiver of an NTP packet and that modifies this NTP packet en route.

   In order to facilitate accurate timestamping, an implementation can
   use a hardware-based timestamping engine, as shown in Figure 1. In
   such cases, NTP packets are sent and received by a software layer,
   whereas a timestamping engine modifies every outgoing NTP packet by
   incorporating its accurate transmission time into the
   <Transmit Timestamp> field in the packet.

```
                         NTP client/server
                    +------------------+
                    |                  |
                    |   +----------+   |
    Software        |   |   NTP    |   |
                    |   | protocol |   |
                    |   +-----+----+   |
                    |         |        |          +----------------------+
                    |   +-----+----+   |         / Intermediate entity  |
                    |   | Accurate |   |        /  in charge of:        |
    ASIC/FPGA       |   | Timestamp|   |      /__  - Timestamping       |
                    |   | engine   |   |       |- Updating checksum or |
                    |   +----------+   |       |  Checksum Complement  |
                    |        |         |       +----------------------+
                    +--------+---------+
                             |
                             |NTP packets
                             |
                        ____ v _
                       /    \_/ \__
                      /           \_
                     /     IP      /
                     \_  Network  /
                      /         \
                      \__/\_  ___/
                           \_/
```
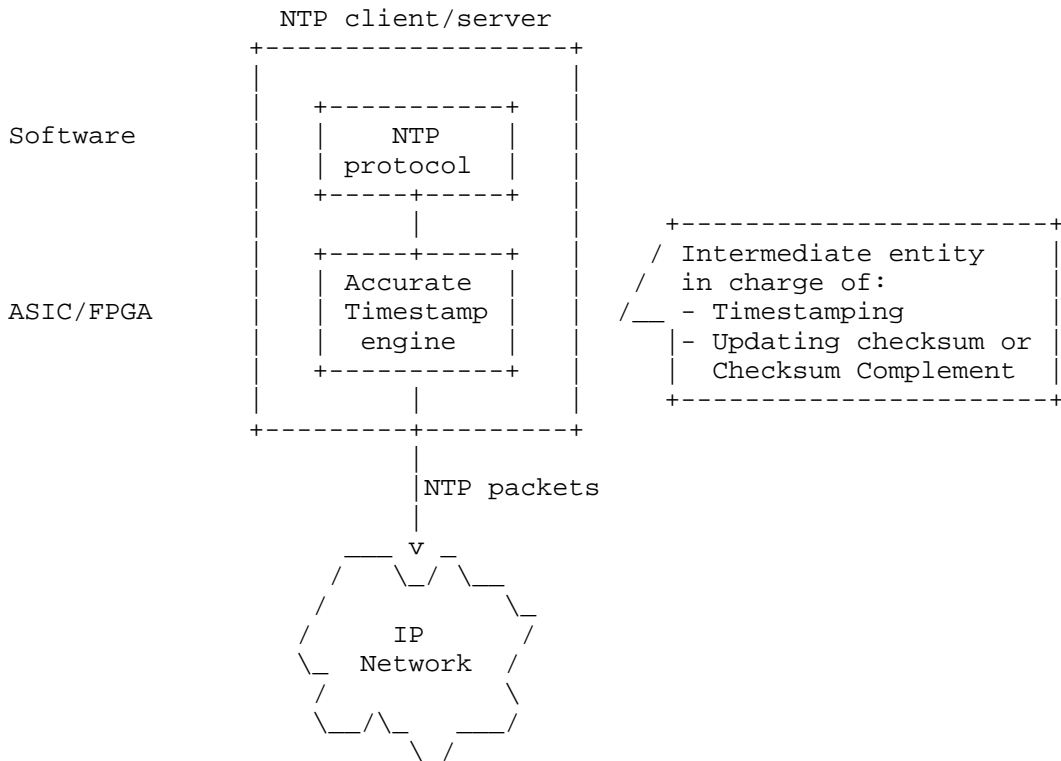
        ASIC: Application-Specific Integrated Circuit
        FPGA: Field-Programmable Gate Array

                 Figure 1: Accurate Timestamping in NTP

   The accuracy of clock synchronization over packet networks is highly
   sensitive to delay jitters in the underlying network; this
   dramatically affects clock accuracy. To address this challenge, the
   Precision Time Protocol (PTP) [IEEE1588] defines Transparent Clocks
   (TCs) -- switches and routers that improve end-to-end clock accuracy
   by updating a "Correction Field" in the PTP packet by adding the
   latency caused by the current TC. In NTP, no equivalent entity is
   currently defined, but future versions of NTP may define an
   intermediate node that modifies en-route NTP packets using a
   "Correction Field".

1.2. Updating the UDP Checksum

   When the UDP payload is modified by an intermediate entity, the UDP
   Checksum field needs to be updated to maintain its correctness. When
   using UDP over IPv4 [UDP], an intermediate entity that cannot update

the value of the UDP Checksum has no choice except to assign a value
of zero to the Checksum field, causing the receiver to ignore the
Checksum field and potentially accept corrupted packets. UDP over
IPv6, as defined in [IPv6], does not allow a zero checksum, except in
specific cases [ZeroChecksum]. As discussed in [ZeroChecksum], the
use of a zero checksum is generally not recommended and should be
avoided to the extent possible.

Since an intermediate entity only modifies a specific field in the
packet, i.e., the Timestamp field, the UDP Checksum update can be
performed incrementally, using the concepts presented in [Checksum].

This document defines the Checksum Complement for [NTPv4]. The
Checksum Complement is a 2-octet field that resides at the end of the
UDP payload. It allows intermediate entities to update NTP packets
and maintain the correctness of the UDP Checksum by modifying the
last 2 octets of the packet, instead of updating the UDP Checksum
field. This is performed by adding an NTP extension field at the end
of the packet, in which the last 2 octets are used as a Checksum
Complement.

The usage of the Checksum Complement can in some cases simplify the
implementation, because if the packet data is processed in serial
order, it is simpler to first update the Timestamp field and then
update the Checksum Complement, rather than to update the timestamp
and then update the UDP Checksum residing at the UDP header. Note
that while it is not impossible to implement a hardware timestamper
that updates the UDP Checksum, using the Checksum Complement instead
can significantly simplify the implementation.

Note that the software layer and the intermediate entity (see
Figure 1) are two modules in a single NTP clock. It is assumed that
these two modules are in agreement regarding whether transmitted NTP
packets include the Checksum Complement or not.

[RFC7820] defines the Checksum Complement mechanism for the One-Way
Active Measurement Protocol (OWAMP) and the Two-Way Active
Measurement Protocol (TWAMP). A similar mechanism is presented in
Annex E of [IEEE1588].

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. Abbreviations

     MAC   Message Authentication Code

     NTP   Network Time Protocol

     PTP   Precision Time Protocol

     UDP   User Datagram Protocol

3. Using the UDP Checksum Complement in NTP

3.1. Overview

   The UDP Checksum Complement is a 2-octet field that is appended at
   the end of the UDP payload, using an NTP extension field. Figure 2
   illustrates the packet format of an NTP packet with a Checksum
   Complement extension.

```
                            +------------------------------+
                            |       IPv4/IPv6 Header        |
                            +------------------------------+
                            |         UDP Header            |
                            +------------------------------+
              ^             |                              |
              |             |                              |
              |             |        NTP packet            |
              |             |                              |
              |             +------------------------------+
            UDP             | Optional NTP Extension Fields |
          Payload           +------------------------------+
              |             |    UDP Checksum Complement    |
              |             |   Extension Field (28 octets) |
              v             +------------------------------+
```
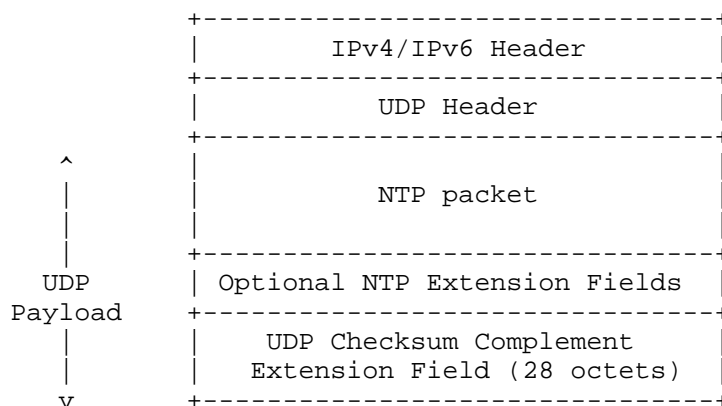
               Figure 2: Checksum Complement in NTP Packets

   The Checksum Complement is used to compensate for changes performed
   in the NTP packet by intermediate entities, as described in the
   Introduction (Section 1). An example of the usage of the Checksum
   Complement is provided in Appendix A.

3.2. Checksum Complement in NTP Packets

   NTP is transported over UDP, either over IPv4 or over IPv6. This
   document applies to both NTP over IPv4 and NTP over IPv6.

   NTP packets may include one or more extension fields, as defined in
   [NTPv4]. The Checksum Complement in NTP packets resides in a
   dedicated NTP extension field, as shown in Figure 3.

   If the NTP packet includes more than one extension field, the
   Checksum Complement extension is always the last extension field.
   Thus, the Checksum Complement is the last 2 octets in the UDP payload
   and is located at (UDP Length - 2 octets) after the beginning of the
   UDP header. Note that the Checksum Complement is not used in
   authenticated NTP packets, as further discussed in Section 3.4.

3.2.1. Using the Checksum Complement

   As described in Section 1, an intermediate entity that updates the
   timestamp in the NTP packet can use the Checksum Complement in order
   to maintain the correctness of the UDP Checksum field. Specifically,
   if the value of the timestamp is updated, this update yields a change
   in the UDP Checksum value; thus, the intermediate entity assigns a
   new value in the Checksum Complement that cancels this change,
   leaving the current value of the UDP Checksum correct. An example of
   the usage of the Checksum Complement is provided in Appendix A.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Field Type           |       Length = 28 octets      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                             MBZ                               |
|                                                               |
|                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |      Checksum Complement       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
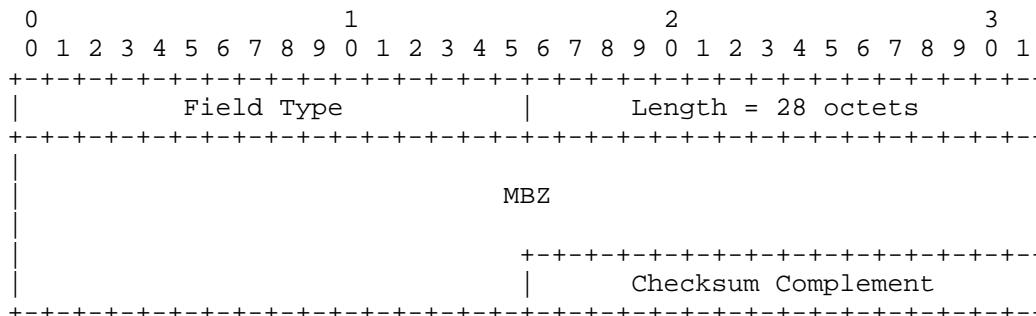
             Figure 3: NTP Checksum Complement Extension Field

Field Type

   A dedicated Field Type value is used to identify the Checksum
   Complement extension. See Section 5.

Length

   The Checksum Complement extension field length is 28 octets.

   This length guarantees that the host that receives the packet
   parses it correctly, whether the packet includes a MAC or not.
   [RFC7822] provides further details about the length of an
   extension field in the absence of a MAC.

MBZ

   The extension field includes a 22-octet MBZ (MUST be zero) field.
   This field MUST be set to 0 and MUST be ignored by the recipient.
   The MBZ field is used for padding the extension field to
   28 octets.

Checksum Complement

   The Checksum Complement extension includes the Checksum Complement
   field, residing in the last 2 octets of the extension.

3.2.2. Transmission of NTP with Checksum Complement

   The transmitter of an NTP packet MAY include a Checksum Complement
   extension field.

3.2.3. Updates of NTP with Checksum Complement

   An intermediate entity that receives and alters an NTP packet
   containing a Checksum Complement extension MAY use the Checksum
   Complement to maintain a correct UDP Checksum value.

3.2.4. Reception of NTP with Checksum Complement

   This document does not impose new requirements on the receiving end
   of an NTP packet.

   The UDP layer at the receiving end verifies the UDP Checksum of
   received NTP packets, and the NTP layer SHOULD ignore the Checksum
   Complement extension field.

3.3. Interoperability with Existing Implementations

   The behavior defined in this document does not impose new
   requirements on the reception of NTP packets beyond the requirements
   defined in [RFC7822]. Note that, as defined in [RFC7822], a host
   that receives an NTP message with an unknown extension field SHOULD
   ignore the extension field and MAY drop the packet if policy requires
   it. Thus, transmitters and intermediate entities that support the
   Checksum Complement can transparently interoperate with receivers
   that are not Checksum Complement compliant, as long as these
   receivers ignore unknown extension fields. It is noted that existing
   implementations that discard packets with unknown extension fields
   cannot interoperate with transmitters that use the Checksum
   Complement.

   It should be noted that when hardware-based timestamping is used, it
   will likely be used at both ends, and thus both hosts that take part
   in the protocol will support the functionality described in this
   memo. If only one of the hosts uses hardware-based timestamping,
   then the Checksum Complement can only be used if it is known that the
   peer host can accept the Checksum Complement.

3.4. The Checksum Complement and Authentication

   A Checksum Complement MUST NOT be used when authentication is
   enabled. The Checksum Complement is useful in unauthenticated mode,
   allowing the intermediate entity to perform serial processing of the
   packet without storing and forwarding it.

   On the other hand, when message authentication is used, an
   intermediate entity that alters NTP packets must also recompute the
   Message Authentication Code (MAC) accordingly. In this case, it is
   not possible to update the Checksum Complement; updating the Checksum
   Complement would result in having to recalculate the MAC, and there
   would be a cyclic dependency between the MAC and the Checksum
   Complement. Hence, when updating the MAC, it is necessary to update
   the UDP Checksum field, making the Checksum Complement field
   unnecessary in the presence of authentication.

4. Security Considerations

   This document describes how a Checksum Complement extension can be
   used for maintaining the correctness of the UDP Checksum.   The
   security considerations of time protocols in general are discussed in
   [SecTime], and the security considerations of NTP are discussed in
   [NTPv4].

   The purpose of this extension is to ease the implementation of
   accurate timestamping engines, as illustrated in Figure 1.   The
   extension is intended to be used internally in an NTP client or
   server.   This extension is not intended to be used by switches and
   routers that reside between the client and the server.   As opposed to
   PTP [IEEE1588], NTP does not require intermediate switches or routers
   to modify the content of NTP messages, and thus any such modification
   should be considered as a malicious man-in-the-middle (MITM) attack.

   It is important to emphasize that the scheme described in this
   document does not increase the protocol's vulnerability to MITM
   attacks; a MITM attacker who maliciously modifies a packet and its
   Checksum Complement is logically equivalent to a MITM attacker who
   modifies a packet and its UDP Checksum field.

   The concept described in this document is intended to be used only in
   unauthenticated mode.   As discussed in Section 3.4, if a
   cryptographic security mechanism is used, then the Checksum
   Complement does not simplify the implementation compared to using the
   conventional Checksum, and therefore the Checksum Complement is not
   used.

5. IANA Considerations

   IANA has allocated a new value in the "NTP Extension Field Types"
   registry:

      0x2005 Checksum Complement

6. References

6.1. Normative References

   [Checksum] Rijsinghani, A., Ed., "Computation of the Internet
               Checksum via Incremental Update", RFC 1624,
               DOI 10.17487/RFC1624, May 1994,
               <http://www.rfc-editor.org/info/rfc1624>.

   [IPv6]   Deering, S. and R. Hinden, "Internet Protocol, Version 6
               (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460,
               December 1998, <http://www.rfc-editor.org/info/rfc2460>.

   [KEYWORDS] 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>.

   [NTPv4]   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>.

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

   [UDP]   Postel, J., "User Datagram Protocol", STD 6, RFC 768,
               DOI 10.17487/RFC768, August 1980,
               <http://www.rfc-editor.org/info/rfc768>.

6.2. Informative References

   [IEEE1588] IEEE, "IEEE Standard for a Precision Clock
               Synchronization Protocol for Networked Measurement and
               Control Systems", IEEE Std 1588-2008,
               DOI 10.1109/IEEESTD.2008.4579760, July 2008.

   [RFC7820]   Mizrahi, T., "UDP Checksum Complement in the One-Way
               Active Measurement Protocol (OWAMP) and Two-Way Active
               Measurement Protocol (TWAMP)", RFC 7820,
               DOI 10.17487/RFC7820, March 2016,
               <http://www.rfc-editor.org/info/rfc7820>.

   [SecTime]   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>.

   [ZeroChecksum]
                Fairhurst, G. and M. Westerlund, "Applicability Statement
                for the Use of IPv6 UDP Datagrams with Zero Checksums",
                RFC 6936, DOI 10.17487/RFC6936, April 2013,
                <http://www.rfc-editor.org/info/rfc6936>.

Appendix A. Checksum Complement Usage Example

   Consider an NTP packet sent by an NTP client to an NTP server.

   The client's software layer (see Figure 1) generates an NTP packet
   with an Origin Timestamp T and a UDP Checksum value U. The value of
   U is the checksum of the UDP header, UDP payload, and pseudo-header.
   Thus, U is equal to:

                     U = Const + checksum(T)   (1)

   Where "Const" is the checksum of all the fields that are covered by
   the checksum, except the Origin Timestamp T.

   Recall that the client's software emits the NTP packet with a
   Checksum Complement extension field, which resides at the end of the
   PTP packet. It is assumed that the client initially assigns zero to
   the value of the Checksum Complement.

   The client's timestamping engine updates the Origin Timestamp field
   to the accurate time, changing its value from T to T'. The engine
   also updates the Checksum Complement field from zero to a new value
   C, such that:

              checksum(C) = checksum(T) - checksum(T')   (2)

   When the NTP packet is transmitted by the client's timestamping
   engine, the value of the checksum remains U as before:

      U = Const + checksum(T) = Const + checksum(T) + checksum(T') -
         checksum(T') = Const + checksum(T') + checksum(C)   (3)

   Thus, after the timestamping engine has updated the timestamp,
   U remains the correct checksum of the packet.

   When the NTP packet reaches the NTP server, the server performs a
   conventional UDP Checksum computation, and the computed value is U.
   Since the Checksum Complement is part of the extension field, its
   value (C) is transparently included in the computation, as per
   Equation (3), without requiring special treatment by the server.

Acknowledgments

Author's Address

   Tal Mizrahi
   Marvell
   6 Hamada St.
   Yokneam, 20692
   Israel

   Email: talmi@marvell.com