Internet Engineering Task Force
Internet-Draft
Obsoletes: 7822 (if approved)
Intended status: Standards Track
Expires: September 27, 2019

H. Stenn
D. Mills
Network Time Foundation

March 26, 2019

Network Time Protocol Version 4 (NTPv4) Extension Fields
draft-stenn-ntp-extension-fields-09

Abstract

Network Time Protocol version 4 (NTPv4) defines the optional usage of extension fields. An extension field, as defined in RFC 5905 [RFC5905] and RFC 5906 [RFC5906], resides after the end of the NTP header and supplies optional capabilities or information that cannot be conveyed in the basic NTP packet. This document updates RFC 5905 [RFC5905] by clarifying some points regarding NTP extension fields and their usage with legacy Message Authentication Codes (MACs), and removes wasteful requirements added by RCF 7822 [RFC7822].

This proposal deprecates RFC 7822 [RFC7822].

RFC EDITOR: PLEASE REMOVE THE FOLLOWING PARAGRAPH BEFORE PUBLISHING:

The source code and issues list for this draft can be found in https://github.com/hstenn/ietf-ntp-extension-fields

Status of This Memo

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

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

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

This Internet-Draft will expire on September 27, 2019.

Table of Contents

1. Introduction 2
2. Conventions Used in This Document 3
2.1. Requirements Language 4
2.2. Terms and Abbreviations 4
3. NTP MAC - RFC 5906 Update 4
3.1. RFC5906 Section 4. - Autokey Cryptography 4
3.2. RFC5906 Section 10. - Autokey Protocol Messages 4
3.3. RFC5906 Section 11.5. - Error Recovery 5
3.4. RFC5906 Section 13. - IANA Consideration 5
4. NTP Extension Fields - RFC 5905 Update 5
4.1. OLD: 'RFC5905 7.5 - NTP Extension Field Format' 5
4.2. NEW: 'RFC5905 Section 7.5 - NTP Extension Field Format' . 6
4.3. NEW: 'RFC5905 Section 7.5.1 - Extension Fields and MACs' 8
4.4. OLD: 'RFC5905 Section 9.2. - Peer Process Operations' . . 10
4.5. NEW: 'RFC5905 Section 9.2. - Peer Process Operations' . . 10
5. Acknowledgements 10
6. IANA Considerations 10
7. Security Considerations 12
8. References 12
8.1. Normative References 12
8.2. Informative References 12
Authors' Addresses 12

1. Introduction

An NTP packet consists of a set of fixed fields that may be followed

by optional fields. Two types of optional fields are defined: extension fields (EFs) as defined in Section 7.5 of RFC 5905 [RFC5905], and legacy Message Authentication Codes (legacy MACs).

If a legacy MAC is used, it resides at the end of the packet. This field can be either a 4-octet crypto-NAK or data that has traditionally been 16, 20 or 24 octets long.

Additional information about the content of a MAC is specified in RFC 5906 [RFC5906], but since that RFC is Informational an implementor that was not planning to provide Autokey would likely never read that document. The result of this would be interoperability problems, at least. To address this problem this proposal also copies and clarifies some of the content of RFC 5906, putting it into RFC 5905. Because there is a reasonable expectation that RFC 5906 will be deprecated, this document does not propose changes or updates to RFC 5906.

NTP extension fields are defined in RFC 5905 [RFC5905] as a generic mechanism that allows the addition of future extensions and features without modifying the NTP header format (Section 16 of RFC 5905 [RFC5905]).

With the knowledge and experience we have gained over time, it has become clear that simplifications, clarifications, and improvements can be made to the NTP specification around EFs and MACs.

This proposal adjusts and clarifies the requirements around EFs and MACs, allows EFs to be on 4-octet boundaries of any acceptable length, and provides methods to disambiguate packet parsing in the unexpected and unlikely case where an implementation would choose to send a packet that could be ambiguously parsed by the receiver.

This proposal deprecates RFC 7822 [RFC7822].

Implementations are still free to send EFs that are padded to longer lengths that otherwise follow the requirements below.

This document better specifies and clarifies extension fields as well as the requirements and parsing of a legacy MAC, with changes to

address errors found after the publication of RFC 5905 [RFC5905] with respect to extension fields. Specifically, this document updates Section 7.5 of RFC 5905 [RFC5905], clarifying the relationship between extension fields and MACs, and expressly defines the behavior of a host that receives an unknown extension field.

2. Conventions Used in This Document

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.2. Terms and Abbreviations

EF - Extension Field

MAC - Message Authentication Code

NTPv4 - Network Time Protocol, Version 4 RFC 5905 [RFC5905]

3. NTP MAC - RFC 5906 Update

This document copies and updates some information in RFC 5906 [RFC5906] and puts it in to RFC 5905, as follows:

3.1. RFC5906 Section 4. - Autokey Cryptography

This section describes some of the cryptography aspects of Autokey. The third paragraph describes the use of 128- and 160-bit message digests. The enumeration of 128- and 160-bit message digests is not meant to be limiting - other message digest lengths MAY be implemented. This paragraph also describes some of the expected semantic ranges of the key ID. This information belongs in RFC 5905. The key ID value is particularly significant because it provides additional detection and disambiguation protection when deciding if

the next data portion is either a legacy MAC or an extension field. [This is additional evidence that although RFC 5906 is Informational, parts of its content are REQUIRED for proper behavior of RFC 5905.]

3.2. RFC5906 Section 10. - Autokey Protocol Messages

This section describes the extension field format, including initial flag bits, a Code field, and 8-bit Field Type, and the 16-bit Length. This proposal expands and clarifies this information and puts it into RFC 5905.

This section says "The reference implementation discards any packet with a field length of more than 1024 characters." but this is no longer true.

3.3. RFC5906 Section 11.5. - Error Recovery

This section describes the crypto-NAK, which should be described in RFC 5905. A crypto-NAK is used by RFC 5905 as well. [This is additional evidence that even though RFC 5906 was Informational, some of its content is REQUIRED for proper behavior for RFC 5095.]

3.4. RFC5906 Section 13. - IANA Consideration

This section lists the Autokey-related Extension Field Types, including Flag Bits, Codes, and Field Types, which should be described in RFC 5905, or perhaps in some other document. [This is additional evidence that even though RFC 5906 is Informational, some of its content is REQUIRED for proper behavior for RFC 5905.]

4. NTP Extension Fields - RFC 5905 Update

This document updates Section 7.5 of RFC 5905 [RFC5905] as follows:

4.1. OLD: 'RFC5905 7.5 - NTP Extension Field Format'

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. Other than defining the field format, this document makes no use of the field contents. An extension field contains a request or response message in the format shown in Figure 14.

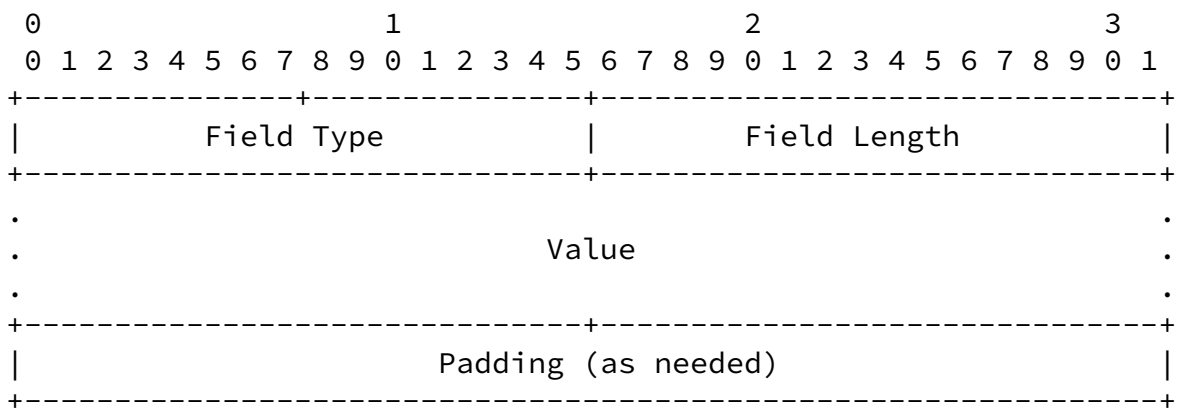


Figure 14: Extension Field Format

All extension fields are zero-padded to a word (four octets) boundary. The Field Type field is specific to the defined function and is not elaborated here. While the minimum field length containing required fields is four words (16 octets), a maximum field length remains to be established.

The Length field is a 16-bit unsigned integer that indicates the length of the entire extension field in octets, including the Padding field.

4.2. NEW: 'RFC5905 Section 7.5 - NTP Extension Field Format'

In NTPv4, one or more extension fields can be inserted after the header and before the possibly optional legacy MAC. A MAC SHOULD be present when an extension field is present. A MAC is always present in some form when NTP packets are authenticated. This MAC SHOULD be either a legacy MAC or a MAC-EF. It MAY be both. Other than defining the field format, this document makes no use of the field contents. An extension field contains a request or response message in the format shown in Figure 14.

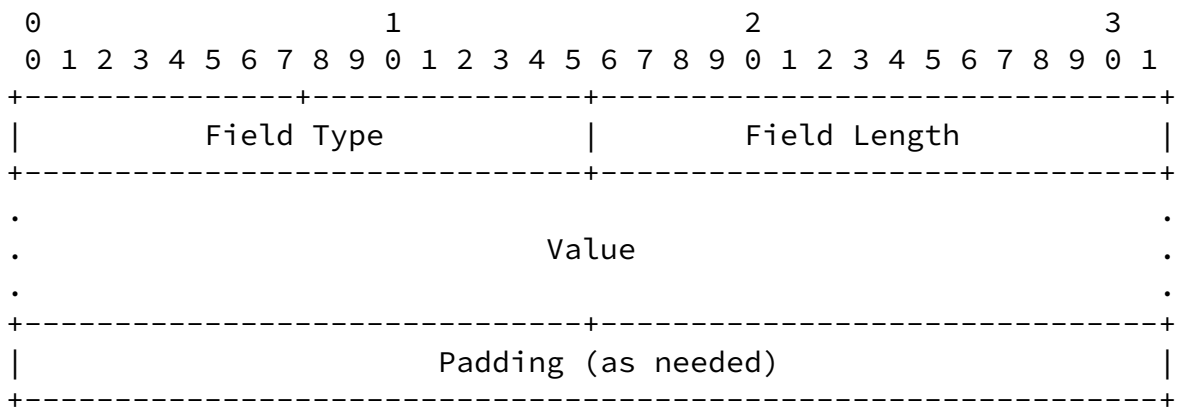


Figure 14: Extension Field Format

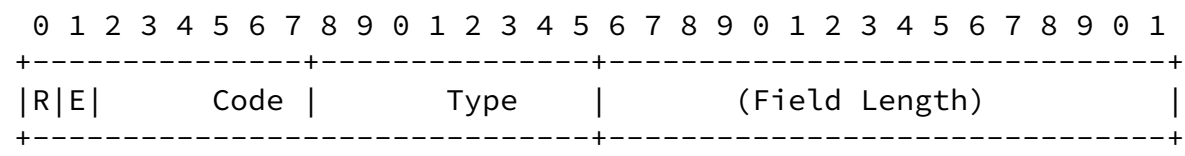
The four octets that comprise the Field Type and Field Length are called the Extension Field Header. Octets beyond the Extension Field Header are called the Extension Field Body, or the Extension Field Payload. The EF Body (EF Payload) MAY be null in some cases.

All extension fields are zero-padded to a word (four octet) boundary. The Field Type is specific to the defined functionality and detailed information about the Field Type is not elaborated here. The minimum size of an Extension Field is a 32-bit word (4 octets), and while the maximum extension field size MUST be 65532 octets or less, an NTP packet SHOULD NOT exceed the network MTU.

The Field Length is a 16-bit unsigned integer that indicates the length of the entire extension field in octets, including any Padding octets. The bottom two bits of the Field Length SHOULD be zero, and the size of the extension field SHOULD end on a 32-bit (4 octet) boundary. [RFC5905 Section 7.5 says "All extension fields are zero-padded to a word (four octets) boundary." but does not use 'MUST' language. Is it overkill to reiterate this requirement here? Should

we use SHOULD or MUST regarding the bottom two bits or the boundary of the EF? It is possible, down the road, that we might find some use for those bottom 2 bits, even if we require a 32-bit boundary on the last octet of an EF.]

The Field Type contains the following sub-elements:



Extension Field Header Format

Where the following Field Type flags are defined:

R: 0 for "Information/Query", 1 for a "Response"

E: 0 for "OK", 1 for an "Error". Unused, and will be deprecated.

[The 'R' flag is currently used by Autokey [RFC5906], and by the proposed I-DO [DRAFT-I-DO] extension field. This flag is used after the packet is accepted.]

[The 'E' flag was proposed for use by Autokey, after the packet was accepted. As it was never used and no other use-cases have been identified, we are recommending this flag be deprecated at some point in the future.]

[The EF Code subtype is currently used by RFC 5906, Autokey [RFC5906], by the proposed Extended Information [DRAFT-EXTENDED-INFORMATION], I-DO [DRAFT-I-DO], and is expected to be used by the NTS Extension Field, at least.]

The Autokey EF currently uses the most Code values - 10 of them, which equates to the least-significant 4 bits of the high-order octet. It is possible that additional flag bits will be allocated; in the past, the high-order 2 bits were reserved, and for a time two additional bits were proposed. Make no assumptions about the unused bits in this octet.

The EF Header and Body fields (the Flags, Code, Type, and Length, and any Value or Padding) are specific to the defined functionality and are not elaborated here; appropriate Field Type Flags, the EF Code, and EF Type values are defined in an IANA registry, and the Length, Value, and Padding values are defined by the document referred to by the registry. If a host receives an extension field with an unknown

Field Type, the host SHOULD ignore the extension field and MAY drop

the packet altogether, depending on local policy.

The Length field is a 16-bit unsigned integer that indicates the length of the entire extension field in octets, including any Padding.

While the minimum field length of an EF that contains no value or padding fields is one word (four octets), and the minimum field length of an EF that contains required fields is two words (8 octets), the maximum field length MUST NOT be longer than 65532 octets due to the maximum size of the data represented by the Length field, and SHOULD be small enough that the size of the NTP packet received by the client does not exceed the smallest MTU between the sender and the recipient. The bottom two bits of the Field Length SHOULD be zero and the EF data SHOULD be aligned to a 32-bit (4 octet) boundary.

4.3. NEW: 'RFC5905 Section 7.5.1 - Extension Fields and MACs'

With the inclusion of additional Extension Fields, there is now a potential that a poorly-designed implementation would produce an ambiguous parsing in the presence of a legacy MAC. What follows are two possibly independent ways to prevent this situation from ever happening.

Note well that to-date, there are only two defined Extension Field Types: Autokey, defined by RFC 5906 [RFC5906], and the Experimental UDP Checksum Complement in the Network Time Protocol, defined by RFC 7821 [RFC7821].

In spite of its known serious problems, Autokey is still in use by some and is a legacy case that is easily supported. Old systems will still work. An old system will still be able to open a properly-configured Autokey association to a new system, a new system will still be able to open a properly-configured Autokey association with an old system, and two new systems will be able to open a properly-configured Autokey association.

The UDP Checksum Complement extension field forbids the use of a legacy MAC, so any packet that uses it CANNOT be using a legacy MAC. [We could list the detailed and specific reasons why traffic using this EF is immune to EF/legacy MAC problems, but I fear that would just be confusing to most people.]

The first and best way to prevent ambiguous parsing is to use the I-DO [DRAFT-I-DO] extension field.

By definition any NTP client or server that handles any other Extension Fields is "new code" and can completely prevent ambiguity by the initiating side sending a packet containing an I-DO [DRAFT-I-DO] extension field followed by an optional MAC-EF [DRAFT-MAC-LAST-EF] followed by an optional legacy MAC. The inclusion of any MAC would be dictated by the authentication requirements of the association.

Note that NTP traffic works perfectly well without using any other extension fields. Newer extension fields offer additional capabilities, but these capabilities are not required for operation. [Even in the case of NTS or SNT, we're talking about "new code" that can be expected to be aware of issues with new extension fields an legacy MACs.]

If the initiating side sends an I-DO [DRAFT-I-DO] packet and gets no response, it operates as if the other side cannot handle new extension fields and simply continues the association without sending any new extension fields. At any point in the future a packet can be sent with an I-DO extension field to see if the other side will respond.

An NTP implementation that receives a packet with an I-DO extension field may respond with a packet that may or may not contain an I-DO Response. If it does not respond, the other side SHOULD assume that the receiver does not understand new EFs. If it responds without sending an I-DO Response extension field, the sending side knows it should not send any new extension fields to this server. If the system that receives an I-DO extension field responds with an I-DO Response, it's telling the sender exactly what capabilities it is currently willing to exchange.

The second way to prevent ambiguous parsing is to use the LAST-EF [DRAFT-MAC-LAST-EF] extension field.

By definition, if I-DO is used and each side agrees to support LAST-EF then LAST-EF will prevent any ambiguity.

If, however, I-DO is not used then one side can simply send a packet with a LAST-EF. The LAST-EF extension field could be four-octet extension field, it could be a 28 octet extension field, or some other length that ends on a 32-bit boundary. If the other side responds appropriately then all is well. If the other side does not respond appropriately the sender should proceed without sending any new extension fields.

Parties interested in additional reasons for and approaches to understanding why there is no reason to be concerned about potential

ambiguities with new code that would use new extension fields and legacy MACs can look at the the drafts that preceded this document.

4.4. OLD: 'RFC5905 Section 9.2. - Peer Process Operations'

...

FXMIT. ... This message includes the normal NTP header data shown in Figure 8, but with a MAC consisting of four octets of zeros. ...

4.5. NEW: 'RFC5905 Section 9.2. - Peer Process Operations'

...

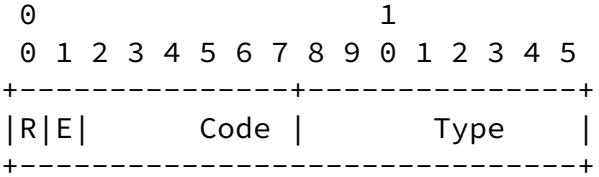
FXMIT. ... This message includes the normal NTP header data shown in Figure 8, but with a MAC consisting of four octets of zeros. This MAC can be a legacy MAC or a MAC-EF. If it's a MAC-EF, the crypto-NAK MUST be the only MAC in the MAC-EF payload. ...

5. Acknowledgements

The authors wish to acknowledge the contributions of Sam Weiler, Danny Mayer, and Tal Mizrahi.

6. IANA Considerations

This memo requests IANA to update the NTP Extension Field Types table in the NTP Parameters document as follows. The following is expected to be a functional superset of the existing information:



NTP Extension Field Type Format

Where the following Field Type flags are defined:

R: 0 for "Information/Query", 1 for a "Response"

E: 0 for "OK", 1 for an "Error". Unused, and will be deprecated.

Field Type	Meaning
0x0000	crypto-NAK (with Field Length of 0)
0x0000	RESERVED: Permanently Unassigned
0x0001	RESERVED: Unassigned
0x0002	Autokey: No-Operation Request
0x8002	Autokey: No-Operation Response
0x0102	Autokey: Association Message Request
0x8102	Autokey: Association Message Response
0x0202	Autokey: Certificate Message Request
0x8202	Autokey: Certificate Message Response
0x0302	Autokey: Cookie Message Request
0x8302	Autokey: Cookie Message Response
0x0402	Autokey: Autokey Message Request
0x8402	Autokey: Autokey Message Response
0x0502	Autokey: Leapseconds Value Message Request
0x8502	Autokey: Leapseconds Value Message Response
0x0602	Autokey: Sign Message Request
0x8602	Autokey: Sign Message Response
0x0702	Autokey: IFF Identity Message Request
0x8702	Autokey: IFF Identity Message Response
0x0802	Autokey: GQ Identity Message Request
0x8802	Autokey: GQ Identity Message Response
0x0902	Autokey: MV Identity Message Request
0x8902	Autokey: MV Identity Message Response
0x0003	* MAC
0x0104	* NTS Unique Identifier Request
0x8104	* NTS Unique Identifier Response
0x0204	* NTS Cookie
0x0304	* NTS Cookie Placeholder
0x0404	* NTS AEEF Request

	0x8404	* NTS AEEF Response
	0x0005	Checksum Complement
	0x2005	Checksum Complement (deprecated flag 0x2000)
	0x0006	* Suggest REFID
	0x0007	* I-DO
	0x0008	* LAST-EF
	0x0009	* Extended Information
	0x00FF	* thru
	0xFDFF	* RESERVED for future I-DO Payloads
	0xFEFF	* I-DO Payload: Leap Smear REFIDs
	0xFFFF	* I-DO Payload: IPv6 REFID hash

* Tentative Reservation

Current Extension Fields

7. Security Considerations

The authors know of no adverse consequences of adopting this proposal.

8. References

8.1. Normative References

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

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

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

[RFC7821] Mizrahi, T., "UDP Checksum Complement in the Network Time

Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March 2016, <https://www.rfc-editor.org/info/rfc7821>.

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

8.2. Informative References

[DRAFT-EXTENDED-INFORMATION]
Stenn, H., "draft-stenn-ntp-extended-information", 2019.

[DRAFT-I-DO]
Stenn, H., "draft-stenn-ntp-i-do", 2019.

[DRAFT-MAC-LAST-EF]
Stenn, H., "draft-stenn-ntp-mac-last-ef", 2019.

Authors' Addresses

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

Email: stenn@nwtime.org

David L. Mills
Network Time Foundation
P.O. Box 918
Talent, OR 97540
US

Email: mills@udel.edu