ETSI TS 103 523-2 V1.1.1 (2021-02)

CYBER;
Middlebox Security Protocol;
Part 2: Transport layer MSP, profile for fine grained access control

Reference
DTS/CYBER-0027-2

Keywords
cyber security

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Cyber Security (CYBER).

The present document is part 2 of a multi-part deliverable covering Middlebox Security Protocols (MSP), defining a generic security blueprint for a family of profiles of MSP, as identified below:

Part 1: "MSP Framework and Template Requirements";

Part 2: "Transport layer MSP, profile for fine grained access control";

Part 3: "Enterprise Transport Security".

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

Requirements exist for network operators, service providers, users, enterprises, and small businesses, to be able to grant varied (fine grained) permissions and to enable visibility of middleboxes, where the middleboxes in turn gain observability of the content and metadata of encrypted sessions. Various cyber defence techniques motivate these requirements. At present, the solutions used often break security mechanisms and/or ignore the desire for explicit authorization by the endpoints. Man-In-The-Middle (MITM) proxies frequently used by enterprises prevent the use of certificate pinning and EV (Extended Validation) certificates. Where no such mechanisms exist, some encryption protocols can even be blocked altogether at the enterprise gateway, forcing users to revert to insecure protocols. As more datagram network traffic is encrypted, the problems for cyber defence will grow (IETF RFC 8404 [i.4]).

The present document is one of a series of implementation profiles to achieve these visibility and observability goals, putting the user in control of the access to their data for cyber defence purposes and protecting against unauthorized access. It sets forth a "Transport layer MSP (TLMSP), profile for fine grained access control" that meets the capability requirements found in Middlebox Security Protocol MSP Part 1 (ETSI TS 103 523-1 [i.5]).

Authorized middleboxes rarely need full read and write access to both the headers and full content of both directions of a communication session to perform their function. TLMSP provides means for classification of the communication between the endpoints into different so-called "contexts", each of which can have different read, delete, and write permissions associated with it, following the security principle of least privilege. This subdivision is for the application to determine and is under endpoint control.

TLMSP is modelled similarly to the TLS protocol (IETF RFC 5246 [1]) and composed of the TLMSP Record Protocol for the encapsulation of data from higher level protocols, and the TLMSP Handshake Protocol for the agreement of keys and the authentication of all parties with access to the communication prior to the sending of any application data. Alert and ChangeCipherSpec Protocols are also provided with similar functionalities as their TLS counterparts. These protocols: satisfy the same basic properties described in IETF RFC 5077 [2], they give visibility and control of the security of the entire communication pathway to the endpoints, and they allow the principle of least privilege to be enforced.

TLMSP is derived from mcTLS [i.1] with added features that include: additional metadata fields that allow middleboxes to perform not only read and modification operations, but also auditable insertions (of new data, originating at the middlebox) and deletions; a more flexible message format, allowing adaptation to varying network conditions; on-path middlebox discovery; improved sequence number handling; fallback to TLS; and additional security measures against recently discovered security vulnerabilities. Three normative annexes are included that contain defined cipher suites, TLS fallback mechanisms, and authentication extensions.

Introduction

There are many uses of middlebox technologies. Some examples are: providing a better user experience (content caching to reduce latency, network prefetching of content); providing user protection and cyber defence (firewalls, intrusion and malware detection, child protection); providing business protection (data loss prevention and audit).

These middlebox systems rarely require both read and write access to all communication content to function, though current security protocols necessitate an all-or-nothing approach, forcing to break the security assurances that underlying encrypted protocols are intended to provide.

EXAMPLE: Man-In-The-Middle proxies used for gateway defence do not provide any assurance of the final endpoint identity, breaking certificate pinning and violating PKI trust models. They also fail to provide assurance that the connection beyond the gateway to the endpoint is even encrypted.

On most non-enterprise networks, users generally desire control of their own data - to choose whether to grant access or not to another party. Users wishing to protect themselves from malicious software on their own systems stealing their data (or including software that harvests user data without user consent) are not currently well-positioned to insist that data is forwarded through their own cyber-defence systems or to grant access to the content. Any system that prevents this can be used as a means of stealing the user data, which is a privacy failure.

To avoid these issues, users need to layer their security architecture and not be forced to rely on endpoint defence alone, as there will be some platforms where this is not optimal, hard, or even impossible. The best defence is always expected to be a layered approach and not reliant on a single mechanism at a single location/layer. This is expected to be particularly true for those low power IoT devices that lack capability of running endpoint protection, where endpoint protection does not even exist, and where patches are slow or non-existent. Unpatched devices can be protected from vulnerabilities only by preventing malicious payloads reaching the IoT device at all; this is a requirement that can only be satisfied by network-based defence.

However, for privacy reasons, network defence ought not to require disabling of data encryption, and maintaining end-to-end encrypted data is a requirement. In the present document, a protocol profile is defined to allow endpoints in a session to authenticate, create an end-to-end encrypted session, and then authorize additional parties to access portions of the encrypted traffic. This profile provides full visibility of all additional middleboxes and their permissions to both parties prior to the sending of any application layer traffic. Additionally, no middleboxes can be added or have permissions granted by this protocol without the both endpoints agreeing to both their presence and their permission level. These requirements assure the fundamental principle that the endpoints are in control of their own data and who can have access to it.

1 Scope

The present document specifies a protocol to enable secure transparent communication sessions between network endpoints with one or more middleboxes between these endpoints, using data encryption and integrity protection, as well as authentication of the identity of the endpoints and the identity of any middlebox present. This protocol can be mapped to the abstract MSP protocol capability requirements in ETSI TS 103 523-1 [i.5].

The Middlebox Security Protocol builds on TLS 1.2 [1] and is an extensively modified version of the mcTLS protocol [i.1]. Whilst basic concepts are inherited from the mcTLS variant, the protocol specified in the present document also contains significant additional functionality and feature changes that would render it incompatible with the original version published.

The present document focuses on TLMSP usage with TCP as it is the most common usage. Usages with other transport protocols are possible but left out of scope. In the remainder of the present document, unless otherwise noted, the word TLS refers to TLS 1.2 [1].

The present document defines a set of five sub-protocols for specific purposes: Handshake (authenticating endpoints and middleboxes and negotiating cryptographic configuration among those entities); Alert (signalling errors and notifications); Application (carrying data generated by higher layers); ChangeCipherSpec (signalling the activation of the negotiated cryptographic configuration) and a Record protocol, (responsible for applying the activated security configuration to all of the other aforementioned sub-protocols).

Since TLMSP is a generic protocol, usable with a wide range of applications, issues related to mapping of application-specific security policy to explicit configurations of TLMSP is largely left out of scope. Further, out-of-band provisioning aspects relating to policies, pre-configuration of the client, details on actions in error situations are also out of scope. While some informal discussion on the security properties of TLMSP is provided, a complete (formal) security analysis of the protocol is currently left out of scope.

A reference implementation of TLMSP is being developed and can be accessed at [i.7].

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] IETF RFC 5246: "The Transport Layer Security (TLS) Protocol Version 1.2".

[2] IETF RFC 5077: "Transport Layer Security (TLS) Session Resumption without Server-side State".

[3] IETF RFC 5116: "An Interface and Algorithms for Authenticated Encryption".

[4] IETF RFC 5746: "Transport Layer Security (TLS) Renegotiation Indication Extension".

[5] IETF RFC 7748: "Elliptic Curves for Security".

[6] IETF RFC 7919: "Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)".

[7] IETF RFC 8449: "Record Size Limit Extension for TLS".

[8] IETF RFC 5288: "AES Galois Counter Mode (GCM) Cipher Suites for TLS".

[9] NIST FIPS PUB 186-4: "Digital Signature Standard (DSS)".

[10] NIST SP 800-38D: "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC".

[11] ETSI TS 133 220: "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA)".

[12] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

[13] IETF RFC 1983: "Internet Users' Glossary".

[14] IETF RFC 1123: "Requirements for Internet Hosts -- Application and Support".

[15] IETF RFC 793: "Transmission Control Protocol".

[16] IETF RFC 791: "Internet Protocol".

[17] IETF RFC 8200: "Internet Protocol, Version 6 (IPv6) Specification".

[18] IEEE 802-2014: "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] D. Naylor et al.: "Multi-Context TLS (mcTLS): Enabling Secure In-Network Functionality in TLS", SIGCOMM '15, August 17 - 21, 2015, London, United Kingdom.

NOTE: http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p199.pdf.

[i.2] D. Naylor: "Architectural Support for Managing Privacy Tradeoffs in the Internet", Carnegie Mellon University, August 2017, PhD Thesis.

NOTE: http://reports-archive.adm.cs.cmu.edu/anon/2017/CMU-CS-17-116.pdf.

[i.3] K. Bhargavan et al.: "A Formal Treatment of Accountable Proxying over TLS", IEEE™ Symposium on Security and Privacy (SP) (2018), May 20 - 24, San Francisco, United States.

[i.4] IETF RFC 8404: "Effects of Pervasive Encryption on Operators".

[i.5] ETSI TS 103 523-1: "CYBER; Middlebox Security Protocol; Part 1: MSP Framework and Template Requirements".

[i.6] D. McGrew, D. Wing, Y. Nir, and P. Gladstone: "TLS Proxy Server Extension", draft-mcgrew-tls-proxy-server-01, IETF.

[i.7] "TLMSP reference implementation".

NOTE: Available at https://forge.etsi.org/rep/cyber.

[i.8] IETF RFC 8446: "The Transport Layer Security (TLS) Protocol Version 1.3".

[i.9] IETF RFC 8447: "IANA Registry Updates for TLS and DTLS".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

1-sided authorization: middlebox traffic observability enabled unilaterally by one endpoint such that the other endpoint is not able to reject or negotiate the traffic observability, other than by ceasing the communication

NOTE: See [i.5].

2-sided authorization: middlebox traffic observability enabled only when both endpoints agree to it

NOTE: See [i.5].

(access) privilege level: per context access rights given to an entity, amongst the four possible options:

- "none" meaning no access rights;
- "read" meaning read access rights only;
- "delete" meaning read and delete access rights only; and
- "write" meaning full access rights - the ability to read, delete, and write (including modify).

NOTE: These access privilege levels are mutually exclusive and each middlebox will have precisely one of the above privilege levels per context.

deleter: for a given context, entity having delete access privilege level with respect to that context

deleter author: for a given context, entity with at least delete access privilege that was the most recent entity to process and forward the message

NOTE 1: Deleter author is considered undefined for contexts when there does not exist any middlebox with explicitly granted delete access.

NOTE 2: TLMSP messages corresponding to context zero never has a deleter author since this context never has explicitly granted delete access.

downstream entity: when sending a TLMSP message in a certain direction, any entity located topologically, relative to the sender, in the direction of the sent message, including the enpoint in that direction

fragment: Service Data Unit (SDU), delivered from one of the higher level TLMSP protocols (Application, Alert, ChangeCipherSpec or Handshake) to the TLMSP Record protocol for protection

(message) author: entity (endpoint or middlebox) making the most recent modification to a message or part thereof

NOTE 1: In TLMSP, there can be up to three distinct authors of a given message. The term author in itself refers to the author of the (possibly encrypted) payload. The other types of authors are the "deleter author" and "writer author", see adjacent definitions. The author, deleter author, and writer author can all be the same entity, or, can all be separate, distinct entities.

NOTE 2: Modification above includes re-encrypting a message using new security parameters of the author, even if the content of the message is unchanged.

(message) originator: entity (endpoint or middlebox) where a new message was first generated and forwarded toward the destination endpoint

NOTE 1: The message originator is invariant. The message author can change as the message is being forwarded.

NOTE 2: The originator and author are only guaranteed to be the same entity at the moment when the message is transmitted by the originator.

reader: for a given context, entity having at least read access privilege level with respect to that context

(TLMSP) context: part of the fragments governed by specific, application dependent access policy

NOTE 1: Here, "part" can refer to a header, a payload, a specific implicitly or explicitly "tagged" part of the payload, or other section of the communication. A special context is defined for non-application data such as handshake and control messages.

NOTE 2: The original mcTLS specification uses the term "slice" instead of "context".

NOTE 3: A context has associated cryptographic keys, made available to those entities that are allowed certain access ("read" and possibly "delete" or "write") to the corresponding context.

(TLMSP) container: order-preserving sub-division of fragments belonging to the Application or Alert protocol, where each sub-division is associated with a specific context or part thereof

(TLMSP) entity: client, server or middlebox engaged in a TLMSP session or the negotiation of such session

(TLMSP) record: Packet Data Unit (PDU) resulting from applying TLMSP security processing directly, either to an entire fragment or to one or more containers, while preserving the inter-container ordering

NOTE: The record is delivered as SDU to lower layer (typically TCP).

upstream entity: when receiving a TLMSP message, any entity located topologically, relative to the receiver, in the direction from which the message is received, including the endpoint in that direction

writer: for a given context, entity having write access privilege level with respect to that context

writer author: for a given context, entity with write access privilege that was the most recent entity to process and forward the message

NOTE: A writer author is always defined and is considered to be the endpoint if no middlebox with write access exists for the given context.

3.2 Symbols

For the purposes of the present document, the following symbols apply:

A || B	concatenation of binary strings A and B
b^n	the n-bit string consisting of the binary value b (0 or 1), repeated n times
B-TID	GBA-defined B-TID value (obtained during GBA bootstrapping)
CTXT_ID	Container Context Identifier
FLAGS	TLMSP container flag field
Ks_NAF	Network Access Function Key
LEN	Length
`ml_d`	Middlebox list, extended by dynamically discovered middleboxes
`ml_i`	Middlebox list (initial)

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

3DES	Triple Data Encryption Standard
3GPP	Third Generation Partnership Project
AAD	Additional Authenticated Data
AEAD	Authenticated Encryption Additional Data
AES	Advanced Encryption Standard
AES-CBC	Advanced Encryption Standard - Cipher Blocker Chaining
AES-GCM	Advanced Encryption Standard - Galois Counter Mode
BSF	Bootstrapping Server Function
CBC	Cipher Block Chaining
CTR	Counter (mode)
DH	Diffie-Hellman
DHE_DSS	Ephemeral Diffie Hellman Digital Signature Standard

DNS	Domain Name System
EV	Extended Validation
FIPS	Federal Information Processing Standard
GBA	Generic Bootstrapping Architecture
GCM	Galois Counter Mode
GMAC	Galois Message Authentication Code
HMAC	Hash-based Message Authentication Code
HTTP	Hypertext Transfer Protocol
IEEE	Institute for Electrical and Electronic Engineers
IoT	Internet of Things
IP	Internet Protocol
IV	Initialization Vector
MAC	Message Authentication Code
MC	Middlebox key Confirmation message
mcTLS	Multi-Context TLS
MITM	Man In The Middle
MK	Middlebox Key material message
MNO	Mobile Network Operator
MSP	Middlebox Security Protocol
NAF	Network Application Function
NAF-Id	Network Application Function Identifier
NAI	Network Access Identifier
NAT	Network Adress Translation
NIST	National Institute of Standards and Technology
PDU	Packet Data Unit
PKI	Public Key Infrastructure
PRF	Pseudorandom Function
RFC	Request for Comments
RSA	Rivest-Shamir-Adleman
SDU	Service Data Unit
SHA	Secure Hash Algorithm
SP	Special Publication
TCAL	TLMSP Context Adaptation Layer
TCP	Transmission Control Protocol
TLMSP	Transport Layer Middlebox Security Protocol
TLS	Transport Layer Security
TR	Technical Report
TS	Technical Specification
USIM	Universal Subscriber Identity Module
UTF	Unicode Transformation Format

4 TLMSP specification

4.1 Introduction

The Transport Layer Middlebox Security Protocol (TLMSP) specified in the present document is derived from the published mcTLS protocol [i.1], [i.2]. The objective is to provide data privacy, data integrity and authentication controls of communication similar to that provided by TLS whilst also providing access to the content (with fine grained access control) to additional authorized and authenticated middleboxes, with visibility of these middleboxes and endpoint control over the permissions granted to middleboxes. Authorized middleboxes rarely need full read and write access to all parts of data and/or to both directions of a communication session to perform their function. TLMSP divides the communication between the endpoints into different contexts, each of which can have different permissions associated with it, following the security principle of least privilege with regards to read and write access. This division of communication is for the application to determine and under endpoint control.

EXAMPLE 1: Application-layer headers and content can be handled as two separate contexts with different associated permissions to each context, described further in annex D.

The TLMSP protocol model builds on the TLS protocol model with a similar presentation language [1]. It is composed mainly of the TLMSP Record Protocol, for the encapsulation of data from higher level TLMSP protocols, and the TLMSP Handshake Protocol, for the agreement of keys and the authentication of all parties with access to the communication prior to the sending of any application data. Alert and ChangeCipherSpec Protocols are also provided with similar functionalities as the TLS counterparts. These protocols satisfy the same basic properties described in the TLS protocol [1]; additionally allowing visibility and control of the security of the entire communication pathway to the endpoints and allowing the principle of least privilege to be enforced.

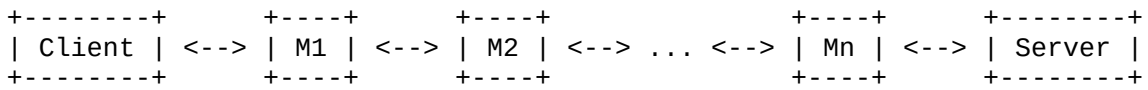


Figure 1: The TLMSP network architecture with client, server and middleboxes M1, M2, …

Unlike the original mcTLS [i.1], the protocol specified here includes:

- additional metadata fields to allow middleboxes to perform not only read and modification operations, but also auditable insertions (of new data, originating at the middlebox) and deletions;
- a more flexible message format, allowing adaptation to varying network conditions;
- on-path middlebox discovery;
- a fallback mechanism to standard TLS; and
- improved robustness of sequence number handling and additional security measures against discovered security vulnerabilities in the original mcTLS specification.

On the topic of TLS-fallback, there could be situations in which a standard TLS client initiates a TLS connection to a server supporting both TLS and TLMSP, but where this server, for whatever reason, has a policy to only allow TLMSP for this particular client. It is out of scope of the present document to specify use-cases for such policies.

EXAMPLE 2: The policy could state that additional 3rd party content filtering is necessary.

4.2 The Record protocol

4.2.1 Overview

4.2.1.1 General

Akin to TLS, the `Record` protocol is a layered protocol that fragments data from higher level protocols (e.g. `Handshake` protocol, `Application` protocol), into TLMSP records, applies the agreed data integrity checks and encryption, and then transmits the resultant records over the transport layer.

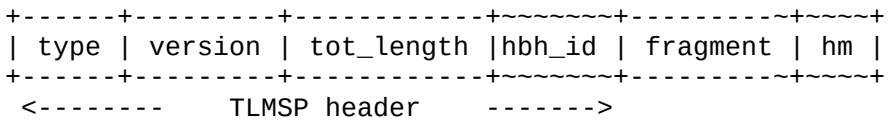
EXAMPLE: TCP can be used for transport. Each TLMSP record delivered to TCP is split across several TCP segments before transmission. Received records (after TCP re-assembly) are decrypted, integrity verified, decompressed, reassembled and then delivered to the higher protocol levels.

The current version of TLMSP does not define or make use of any (non-trivial) compression method, due to several foreseen issues as discussed in annex H. Future versions of TLMSP may specify usage of compression.

4.2.1.2 Records, containers and contexts

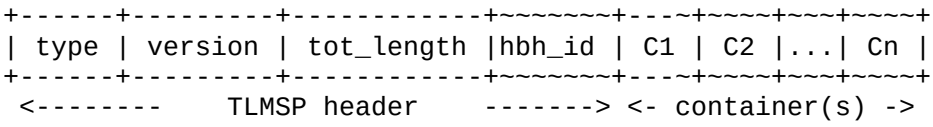
For TLMSP to allow the traffic optimizations it seeks to enable, TLMSP allows data fragments associated with multiple contexts to be "packaged" into one single TLMSP record and also allows for data associated with a single context to be split across records. Thus, a *TLMSP record* comprises protected data corresponding to one or more *TLMSP contexts*. Within a record, a (contiguous) fragment of data associated with a context is called a *TLMSP container* (or simply *container*). An explicit container format shall be used for the `Alert` and `Application` protocols, but not for the `Handshake` and `ChangeCipherSpec` protocols, both of which are associated with a default context called *context zero*.

4.2.1.3 Record and container construction and processing overview



NOTE: The field `hm` is the hop-by-hop MAC and is present only for Handshake records occurring after ChangeCipherSpec.

Figure 2a: TLMSP record format not using containers used by the Handshake and ChangeCipherSpec protocol



NOTE: C1, C2, … Cn represents containers, whose format is defined in Figure 3.

Figure 2b: TLMSP record format using containers (as used by Application and Alert protocols after server confirmation of TLMSP support)

The first five octets of the TLMSP header comprising `type`, `version`, and `tot_length` shall be formatted as a TLS 1.2 header as per clause 6.2.1 of IETF RFC 5246 [1].

EXAMPLE 1: `type` = 0x15 is used to signal the `Alert` protocol.

In the `ServerHello`, confirming TLMSP extension support, and in all records thereafter, there shall after the `tot_length` field follow the `hbh_id` field which is a variable length (possibly zero length) identifier for the TLMSP session, valid on a particular hop (between neighbouring entities). The `hbh_id` shall be chosen by the transmitting entity for each hop as defined in clause 4.3.5 and shall be used as defined in clauses 4.2.2.1 and 4.3.5.

The field `tot_length` shall define the total (octet) length of the record following the `tot_length` field itself, i.e. including the length indicator portion of `hbh_id` plus the indicated number of octets (which may be zero). TLMSP allows record lengths up to 2^{16} -1. However, if a TLMSP client is willing to accept lengths above the normal IETF RFC 5246 maximum of 2^{14} octets [1], this shall be signalled using the extension of IETF RFC 8449 [7]. The server and middleboxes, observing the client extension may accept or limit the length by including their corresponding maximum acceptable lengths in their extensions. The maximum length to be used shall be the minimum over the lengths occurring in all entities' extensions.

After the TLMSP record header, there shall follow the actual container(s) for those TLMSP protocols that use containers, i.e. `Alert` and `Application`. For all other TLMSP protocols, a single fragment shall follow (see clause 4.2.7.1 for details). When record protection is active, all protocols except `ChangeCipherSpec` shall then include a hop-by-hop MAC tag, denoted `hm` and computed according to clause 4.2.7.2.3, added at the end of the record in order to integrity protect the entire record (excluding `hm` itself).

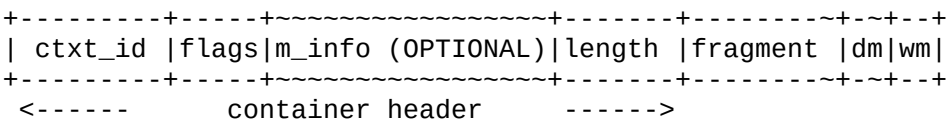


Figure 3: TLMSP container format

A container consists of a header, a (data) fragment (including a reader MAC) and one or two additional MAC values, `dm` (conditionally optional), and `wm`. Specifically, each container shall start with a container header which shall include all of the following: the associated one-octet context identifier `ctxt_id` (where `ctxt_id` = 0 is reserved), two bytes reserved for `flags`, and a 16-bit `length` field, indicating the length up to the end of the `fragment` field.