Network Working Group
Internet-Draft
Obsoletes: 6083 (if approved)
Intended status: Standards Track
Expires: 23 October 2025

M. T端xen
M端nster Univ. of Applied Sciences
H. Tschofenig

T. Reddy
Nokia

21 April 2025

Datagram Transport Layer Security (DTLS) 1.3 for Stream Control Transmission Protocol (SCTP)
draft-tuexen-tsvwg-rfc6083-bis-07

Abstract

This document describes the usage of the Datagram Transport Layer Security (DTLS) 1.3 protocol over the Stream Control Transmission Protocol (SCTP) and obsoletes RFC 6083.

DTLS 1.3 over SCTP provides communications privacy for applications that use SCTP as their transport protocol and allows client/server applications to communicate in a way that is designed to prevent eavesdropping and detect tampering or message forgery.

Applications using DTLS 1.3 over SCTP can use almost all transport features provided by SCTP and its extensions.

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 23 October 2025.

Copyright Notice

Table of Contents

1. Introduction

This document describes the usage of the Datagram Transport Layer Security (DTLS) 1.3 protocol, as defined in [RFC9147], over the Stream Control Transmission Protocol (SCTP), as defined in [RFC9260].

Prior versions of DTLS are out of scope for this document. The use of DTLS 1.0 is described in [RFC6083]. For the rest of the document we assume version 1.3 when referring to DTLS unless the context requires it to refer to a dedicated version.

DTLS over SCTP provides communications privacy for applications that use SCTP as their transport protocol and allows client/server applications to communicate in a way that is designed to prevent eavesdropping and detect tampering or message forgery.

Applications using DTLS over SCTP can use almost all transport features provided by SCTP and its extensions.

TLS is designed to run on top of a byte-stream-oriented transport protocol providing a reliable, in-sequence delivery.

TLS over SCTP, as described in [RFC3436], has limitations:

* It does not support the unordered delivery of SCTP user messages.

* It does not support partial reliability, as defined in [RFC3758].

* It only supports the usage of the same number of streams in both directions.

* It uses a TLS connection for every bidirectional stream, which requires a substantial amount of resources and message exchanges if a large number of streams is used.

DTLS over SCTP, as described in this document, overcomes these limitations of TLS over SCTP. This specification supports:

* preservation of message boundaries.

* a large number of unidirectional and bidirectional streams.

* ordered and unordered delivery of SCTP user messages.

* the partial reliability extension, as defined in [RFC3758].

* the dynamic address reconfiguration extension, as defined in [RFC5061].

The list above matches the design of DTLS over SCTP based on [RFC6083].

This specification supports two ways of relaxing the message size limitation, which is imposed by the maximum plaintext record size of 2^14 bytes:

* Using DTLS extensions allows to bump this limit to 2^32 - 256 bytes.

* If only ordered and reliable messages are relevant, remove the limit at all by using a fragmentation and reassembly method making use of the Payload Protocol Identifier (PPID).

However, the following limitation still apply:

* The DTLS user cannot perform the SCTP-AUTH key management because this is done by the DTLS layer.

DTLS establishes session keys for SCTP-AUTH in the same style as [RFC5763] defines how DTLS establishes keys for SRTP. This specification utilizes a new version of SCTP-AUTH, described in [I-D.ietf-tsvwg-rfc4895-bis] utilizing modern cryptographic algorithms.

The method described in this document requires that the SCTP implementation supports the optional feature of fragmentation of SCTP user messages as defined in [RFC9260] and the SCTP authentication extension defined in [I-D.ietf-tsvwg-rfc4895-bis].

The design of this specification is based on the following principles:

* Re-use RFC 6083 as much as possible. Large parts of RFC 6083 are re-used.

* Focus of DTLS 1.3 and use its features and extensions.

* Minimize the implementation effort.

* Re-use the SCTP-AUTH extension but utilize a modernized design.

2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the following terms:

Association: An SCTP association.

Stream: A unidirectional stream of an SCTP association. It is uniquely identified by a stream identifier.

This document uses the following terms:

DTLS: Datagram Transport Layer Security

MTU: Maximum Transmission Unit

PPID: Payload Protocol Identifier

SCTP: Stream Control Transmission Protocol

TCP: Transmission Control Protocol

TLS: Transport Layer Security

3. DTLS 1.3 Considerations

3.1. Version of DTLS

This document is based on [RFC9147].

3.2. Message Sizes

DTLS, as specified in [RFC9147], limits the maximum plaintext record size to 2^14 bytes.

If this limit is too restrictive, there are at least two options:

1. To bump the maximum plaintext record size limit to 2^32 - 256 bytes, the "record_size_limit" extension, which is defined in [RFC8449], MAY be used in combination with [I-D.ietf-tls-tlsflags] and [I-D.mattsson-tls-super-jumbo-record-limit].

2. If only ordered and reliable messages are relevant, the PPID based fragmentation and reassembly method specified in [I-D.tuexen-tsvwg-sctp-ppid-frag] MAY be used to remove the limit at all.

3.3. Replay Detection

The DTLS protocol allows two forms of replay protection: replay detection of handshake messages and replay detection of application data payloads. Handshake messages must be reliably transmitted and replay detection is essential.

Contrary to handshake messages, detecting replays of application data messages is optional. If enabled, this replay detection may result in the DTLS layer dropping messages. Since DTLS/SCTP provides a reliable service, if requested by the application, replay detection cannot be used. Therefore, replay detection for application data payloads of DTLS MUST NOT be used.

3.4. Path MTU Discovery

SCTP provides Path MTU discovery and fragmentation/reassembly for user messages. Since DTLS can send maximum sized messages Path MTU discovery of DTLS MUST NOT be used.

Still, DTLS cannot completely ignore the PMTU for reasons mentioned in Section 4.4 of [RFC9147]. The DTLS record framing expands the datagram size, thus lowering the effective PMTU.

As recommended in Section 4.4 of [RFC9147], the DTLS record layer SHOULD also allow the upper layer protocol to discover the amount of record expansion expected by the DTLS processing.

3.5. Retransmission of Messages

SCTP provides a reliable and in-sequence transport service for DTLS messages that require it. Therefore, DTLS procedures for retransmissions of handshake messages MUST NOT be used. For the DTLS stack the appearance is that there is no message loss and consequently no retransmission needed.

3.6. Exporter

TLS defines an exporter interface, which is inherited by DTLS. The exporter interface allows the application using DTLS to obtain keying material from the DTLS stack. In [RFC8446] the exporter values are computed as:

```
TLS-Exporter(label, context_value, key_length) =
    HKDF-Expand-Label(Derive-Secret(Secret, label, ""),
                      "exporter", Hash(context_value), key_length)
```

For use with this specification the label MUST be set to "EXPORTER_DTLS13_OVER_SCTP", an empty context_value field and a key length of 64 bytes.

3.7. Application Message Length

The size of a DTLS record header depends on several factors, namely

* the use of the DTLS Connection ID (CID) feature,

* the size of the sequence number,

* the presence of the length field, and

* the use of padding to conceal the true message length.

While the CID needs to be negotiated with the peer, the other parameters can be configured in DTLS stacks. The size of the records can, however, be influenced with the record size limit extension and the flags extension.

Hence, an SCTP user application has a number of configuration options to adjust the use of DTLS to best fit a given deployment environment.

4. SCTP Considerations

4.1. Mapping of DTLS Records

The supported maximum length of SCTP user messages MUST be at least the size of the maximum size of a DTLSCiphertext. In particular, the SCTP implementation MUST support fragmentation of user messages.

Every SCTP user message MUST consist of exactly one DTLS record.

4.2. DTLS Connection Handling

Each DTLS connection MUST be established and terminated within the same SCTP association. A DTLS connection MUST NOT span multiple SCTP associations.

4.3. Payload Protocol Identifier Usage

Application protocols using DTLS over SCTP SHOULD register and use a separate payload protocol identifier (PPID) and SHOULD NOT reuse the PPID that they registered for running directly over SCTP.

Using the same PPID does not harm as long as the application can determine whether or not DTLS is used. However, for protocol analyzers, for example, it is much easier if a separate PPID is used.

This means, in particular, that there is no specific PPID for DTLS.

4.4. Stream Usage

All DTLS messages except ApplicationData protocol messages MUST be transported on stream 0 with unlimited reliability and with the ordered delivery feature.

DTLS messages of the ApplicationData protocol SHOULD use multiple streams other than stream 0; they MAY use stream 0 for everything if they do not care about minimizing head of line blocking.

4.5. Chunk Handling

DATA chunks of SCTP MUST be sent in an authenticated way, as described in [I-D.ietf-tsvwg-rfc4895-bis]. Other chunks MAY be sent in an authenticated way. This makes sure that an attacker cannot modify the stream in which a message is sent or affect the ordered/unordered delivery of the message.

If PR-SCTP, as defined in [RFC3758], is used, FORWARD-TSN chunks MUST also be sent in an authenticated way, as described in [I-D.ietf-tsvwg-rfc4895-bis]. Hence, it is not possible for an attacker to drop messages and use forged FORWARD-TSN, SACK, and/or SHUTDOWN chunks to hide this dropping.

4.6. Post-Handshake Authentication

If the SCTP peers require periodic re-authentication after DTLS handshake is complete to provide proof of ownership of an updated identity (e.g., X.509 certificate), the mechanism defined in [RFC9261] MUST be leveraged by the peers for mutual re-authentication. The application-layer protocol would have to be used to send the authenticator request, either by the SCTP client or the SCTP server using the established DTLS connection.

A specific PPID is used to multiplex/de-multiplex user messages used for performing the procedures defined in [RFC9261] with other user messages.

4.7. Rekeying

The Extended Key Update mechanism for DTLS 1.3, defined in [I-D.ietf-tls-extended-key-update], specifies the ExtendedKeyUpdate message, which indicates that the sender wishes to update its cryptographic keys to ensure forward secrecy. In DTLS, updating traffic keys requires the epoch value to be incremented. Since epoch values cannot wrap, the maximum number of epoch updates is 2^64 (this includes epoch updates during the handshake itself).

When the sender of the ExtendedKeyUpdate message receives an ExtendedKeyUpdateResponse, it knows that it can safely switch from the old epoch to the new epoch once the NewKeyUpdate message has been successfully transmitted. This process is described in Section 8 of [RFC9147] and Section 8 of [I-D.ietf-tls-extended-key-update]. While DTLS 1.3 will automatically perform this switch, the SCTP application using DTLS 1.3 for updating keys with SCTP-AUTH requires updating of keying material. Although it is not mandatory to synchronize SCTP-AUTH key updates with DTLS key changes, it is RECOMMENDED to switch keys for use with SCTP-AUTH once a NewKeyUpdate message has been successfully transmitted.

4.8. Handshake

A DTLS implementation discards DTLS messages from older epochs after some time, as described in [RFC9147]. This is not acceptable when a reliable data transfer is performed.

4.9. Handling of Endpoint-Pair Shared Secrets

The endpoint-pair shared secret for Shared Key Identifier 0 is empty and MUST be used when establishing a DTLS connection. A new endpoint-pair shared secret MUST be established using the exporter interface defined in [RFC8446], as described in Section 3.6. The new Shared Key Identifier MUST be the old Shared Key Identifier incremented by 1. If the old one is 65535, the new one MUST be 1.

Before sending the Finished message, the active SCTP-AUTH key MUST be switched to the new one.

Once the corresponding Finished message from the peer has been received, the old SCTP-AUTH key SHOULD be removed.

Since ExtendedKeyUpdate messages are used to change the application traffic keys it is necessary to update the keying material initially exported via the TLS 1.3 exporter interface.

The key derivation function MUST be used once the NewKeyUpdate has been transmitted successfully. The function follows the design of the exporter interface and deriving the updated Exported Keying Material is discussed in Section 11 of [I-D.ietf-tls-extended-key-update].

For use with this specification the label MUST be set to "DERIVE_DTLS13_OVER_SCTP", a context_value field containing a 64 bit sequence number and a key length of 64 bytes. The 64 bit number logically corresponds to the epoch value but there is no requirement for the DTLS stack to expose the epoch value to this key derivation function interfacing the SCTP stack for setting the SCTP-AUTH keying material. The sequence number MUST be increased with every DTLS key update.

4.10. Shutdown

To prevent DTLS from discarding DTLS user messages while it is shutting down, a close_notify alert MUST only be sent after all outstanding SCTP user messages have been acknowledged by the SCTP peer and MUST NOT still be revoked by the SCTP peer.

Prior to processing a received close_notify, all other received SCTP user messages that are buffered in the SCTP layer MUST be read and processed by DTLS.

5. IANA Considerations

IANA is asked to update in the the TLS Exporter Label registry, which was established in [RFC5705] and updated by [RFC8447], the Reference of the label "EXPORTER_DTLS_OVER_SCTP" to this document.

6. Security Considerations

The security considerations given in [RFC4347], [RFC4895], and [RFC9260] also apply to this document.

It is possible to authenticate DTLS endpoints based on IP addresses in certificates. SCTP associations can use multiple addresses per SCTP endpoint. Therefore, it is possible that DTLS records will be sent from a different IP address than that originally authenticated. This is not a problem provided that no security decisions are made based on that IP address. This is a special case of a general rule: all decisions should be based on the peer's authenticated identity, not on its transport layer identity.

For each message, the SCTP user also provides a stream identifier, a flag to indicate whether the message is sent ordered or unordered, and a payload protocol identifier. Although DTLS can be used to provide privacy for the actual user message, none of these three are protected by DTLS. They are sent as clear text, because they are part of the SCTP DATA chunk header.

While TLS 1.3 reduced the number of supported cipher suites and removed a number of cipher suites, including all NULL cipher algorithm. However, [RFC9150] later re-introduced support for cipher suites that do not support confidentiality protection. Negotiating such cipher suites will not provide communications privacy for SCTP applications and will not provide privacy for user messages and MUST NOT be used with this specification.

7. Acknowledgments

The authors wish to thank Eric Rescorla and Robin Seggelmann for being coauthors of [RFC6083], which is the basis of this document.

The authors wish to thank Anna Brunstrom, Lars Eggert, Gorry Fairhurst, Ian Goldberg, Alfred Hoenes, Carsten Hohendorf, Stefan Lindskog, Daniel Mentz, and Sean Turner for their invaluable comments on [RFC6083].

Finally, the authors wish to thank Eric Rescorla and Martin Thomson for their invaluable comments on this document.

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>.

[RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, DOI 10.17487/RFC3758, May 2004, <https://www.rfc-editor.org/info/rfc3758>.

[RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, DOI 10.17487/RFC4347, April 2006, <https://www.rfc-editor.org/info/rfc4347>.

[RFC4895] Tuexen, M., Stewart, R., Lei, P., and E. Rescorla, "Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)", RFC 4895, DOI 10.17487/RFC4895, August 2007, <https://www.rfc-editor.org/info/rfc4895>.

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

[RFC5763] Fischl, J., Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, DOI 10.17487/RFC5763, May 2010, <https://www.rfc-editor.org/info/rfc5763>.

[RFC8449] Thomson, M., "Record Size Limit Extension for TLS", RFC 8449, DOI 10.17487/RFC8449, August 2018, <https://www.rfc-editor.org/info/rfc8449>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

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

[RFC8447] Salowey, J. and S. Turner, "IANA Registry Updates for TLS and DTLS", RFC 8447, DOI 10.17487/RFC8447, August 2018, <https://www.rfc-editor.org/info/rfc8447>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/info/rfc9147>.

[RFC9260] Stewart, R., T端xen, M., and K. Nielsen, "Stream Control Transmission Protocol", RFC 9260, DOI 10.17487/RFC9260, June 2022, <https://www.rfc-editor.org/info/rfc9260>.

[RFC9261] Sullivan, N., "Exported Authenticators in TLS", RFC 9261, DOI 10.17487/RFC9261, July 2022, <https://www.rfc-editor.org/info/rfc9261>.

[I-D.ietf-tsvwg-rfc4895-bis]
T端xen, M., Stewart, R. R., Lei, P., and H. Tschofenig, "Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)", Work in Progress, Internet-Draft, draft-ietf-tsvwg-rfc4895-bis-04, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-tsvwg-rfc4895-bis-04>.

[I-D.ietf-tls-tlsflags]
Nir, Y., "A Flags Extension for TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-tlsflags-15, 15 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-tlsflags-15>.

[I-D.mattsson-tls-super-jumbo-record-limit]
Mattsson, J. P., Tschofenig, H., and M. T端xen, "Large Record Sizes for TLS and DTLS with Reduced Overhead", Work in Progress, Internet-Draft, draft-mattsson-tls-super-jumbo-record-limit-05, 5 September 2024, <https://datatracker.ietf.org/doc/html/draft-mattsson-tls-super-jumbo-record-limit-05>.

[I-D.tuexen-tsvwg-sctp-ppid-frag]
T端xen, M., Jesup, R., and H. Tschofenig, "Payload Protocol Identifier based Fragmentation and Reassembly for the Stream Control Transmission Protocol", Work in Progress, Internet-Draft, draft-tuexen-tsvwg-sctp-ppid-frag-03, 2 March 2025, <https://datatracker.ietf.org/doc/html/draft-tuexen-tsvwg-sctp-ppid-frag-03>.

[I-D.ietf-tls-extended-key-update]
Tschofenig, H., T端xen, M., Reddy.K, T., Fries, S., and Y. Rosomakho, "Extended Key Update for Transport Layer Security (TLS) 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-extended-key-update-02, 20 October 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-extended-key-update-02>.

8.2. Informative References

[RFC3436] Jungmaier, A., Rescorla, E., and M. Tuexen, "Transport Layer Security over Stream Control Transmission Protocol", RFC 3436, DOI 10.17487/RFC3436, December 2002, <https://www.rfc-editor.org/info/rfc3436>.

[RFC5061] Stewart, R., Xie, Q., Tuexen, M., Maruyama, S., and M. Kozuka, "Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration", RFC 5061, DOI 10.17487/RFC5061, September 2007, <https://www.rfc-editor.org/info/rfc5061>.

[RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, DOI 10.17487/RFC6083, January 2011, <https://www.rfc-editor.org/info/rfc6083>.

[RFC9150] Cam-Winget, N. and J. Visoky, "TLS 1.3 Authentication and Integrity-Only Cipher Suites", RFC 9150, DOI 10.17487/RFC9150, April 2022, <https://www.rfc-editor.org/info/rfc9150>.

Authors' Addresses

Michael T端xen
M端nster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
Germany
Email: tuexen@fh-muenster.de

Hannes Tschofenig
Email: hannes.tschofenig@gmx.net

Tirumaleswar Reddy
Nokia
Email: k.tirumaleswar_reddy@nokia.com