Transport and Services Working Group
Internet-Draft
Intended status: Informational
Expires: 2 July 2026

P. Hancke
Meta Platforms Inc.
J. Uberti
OpenAI
V. Boivie
Google

29 December 2025

SCTP Negotiation Acceleration Protocol
draft-hancke-tsvwg-snap-00

Abstract

WebRTC Data Channels use the Stream Control Transmission Protocol (SCTP) over a Datagram Transport Layer Security (DTLS) association. The standard SCTP connection establishment requires a handshake that introduces latency. This document specifies a method to accelerate the datachannel establishment by embedding the SCTP initialization parameters within the Session Description Protocol (SDP) offer/answer exchange. This reduces the time required to open a data channel by up to two network round-trip times.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://fippo.github.io/warp-snap-sped/draft-hancke-tsvwg-snap-latest.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-hancke-tsvwg-snap/.

Discussion of this document takes place on the WG Working Group mailing list (mailto:tsvwg@ietf.org), which is archived at https://datatracker.ietf.org/wg/tsvwg/. Subscribe at https://www.ietf.org/mailman/listinfo/tsvwg/.

Source for this draft and an issue tracker can be found at https://github.com/fippo/warp-snap-sped.

Status of This Memo

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

This Internet-Draft will expire on 2 July 2026.

Copyright Notice

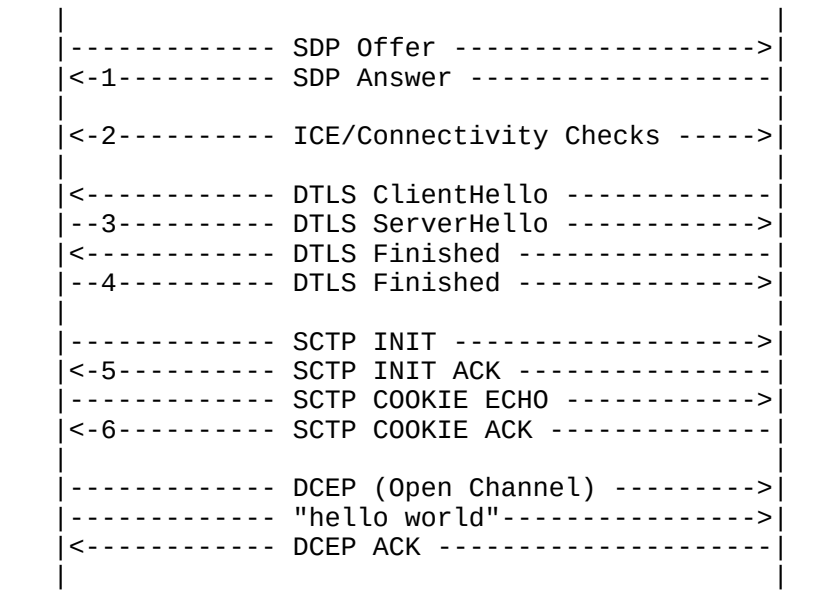
Table of Contents

1. Introduction

[RFC8831] defines WebRTC Data Channels that allow the transport of arbitrary non-media data over a WebRTC PeerConnection. This uses SCTP [RFC9260] and a DTLS encapsulation of SCTP packets [RFC8261].

SCTP establishes its associations using a four-way handshake, which primarily serves to protect against half-open (SYN-flood) attacks. For WebRTC, SCTP runs encapsulated within DTLS [RFC8261], which establishes a secure, encrypted channel between the peers that prevents half-open attacks.

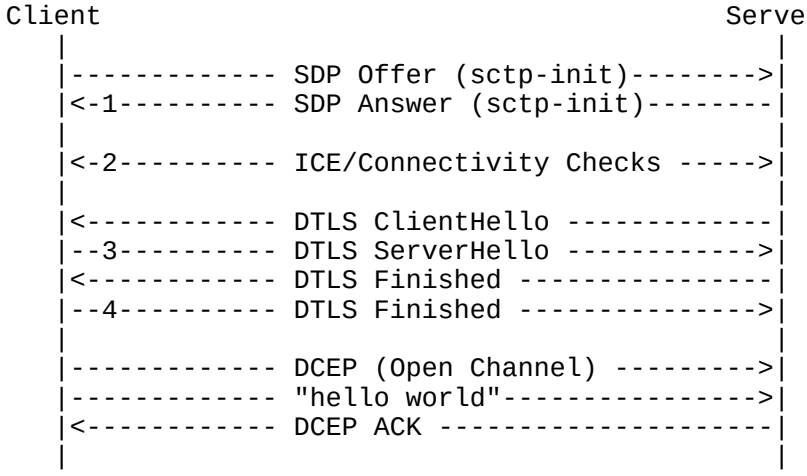
The full connection setup between two entities using WebRTC, consisting of the exchange of SDP over the signaling channel, ICE and DTLS establishment and the SCTP handshake, is shown below (with annotations for the number of full round trips):



Note that SCTP typically does a "simultaneous open", i.e. both sides take the active role, although only one direction is shown here for simplicity.

Note: [RFC9260] allows the packets with the COOKIE ECHO and COOKIE ACK to carry data. This means the second RTT is not strictly necessary but was observed in practice.

When the SCTP INIT chunk is signaled in the SDP, using the "sctp-init" attribute described in this document, this flow can be reduced to the following exchange:



2. Conventions and Definitions

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.

3. Terminology

SCTP INIT: the SCTP INIT chunk as described in Section 3.3.2 of [RFC9260].

SCTP INIT ACK: the SCTP INIT ACK chunk as described in Section 3.3.3 of [RFC9260].

SCTP COOKIE ECHO: the SCTP COOKIE ECHO chunk as described in Section 3.3.11 of [RFC9260].

SCTP COOKIE ACK: the SCTP COOKIE ACK chunk as described in Section 3.3.12 of [RFC9260].

RTT: Round-Trip Time

4. SDP sctp-init attribute

4.1. General

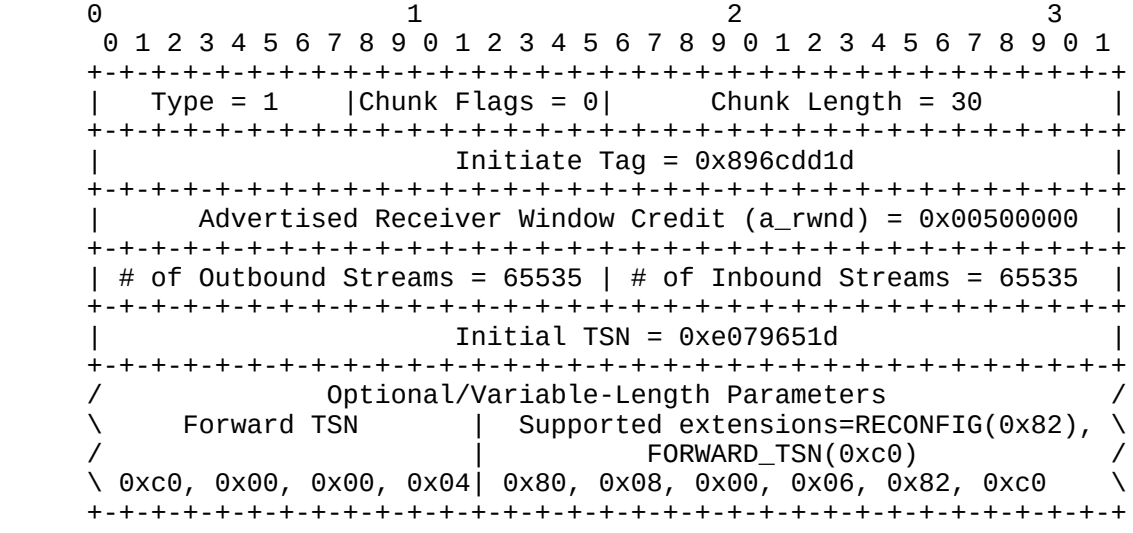
This section defines a new SDP media-level attribute, "sctp-init". The attribute can be associated with an SDP media description ("m=" line) with a "UDP/DTLS/SCTP" protocol identifier (defined in [RFC8841]) and the <fmt> parameter value of 'webrtc-datachannel' (defined in [RFC8832]).

NOTE: This specification only defines the usage of the SDP "sctp-init" attribute when associated with an "m=" line containing one of the following proto values: "UDP/DTLS/SCTP" or "TCP/DTLS/SCTP". Usage of the attribute with other proto values needs to be defined in a separate specification.

An example follows, see Section 7 for the full SDP exchange:

a=sctp-init:AQAAHols3R0AUAAA/////+B5ZR3AAAAEgAgABoLA

This is equivalent to the following SCTP INIT chunk:



4.2. Syntax

Attribute name: sctp-init

Type of attribute: media

Mux category: CAUTION

Subject to charset: No

Purpose: allows the SDP to carry the information contained in a SCTP INIT chunk

Appropriate values: A base64-encoded value.

Syntax: sctp-init-value = base64 ; base64 defined in RFC 4566

5. SDP Offer/Answer Procedures

5.1. General

This section defines how the "sctp-init" attribute can be negotiated using the SDP offer/answer mechanism.

5.2. Generating the Initial SDP Offer

If the offering endpoint has created a WebRTC data channel, it MUST let the SCTP implementation generate a serialized INIT chunk that it would normally send over the network as a series of bytes. This message is then base64-encoded and added to the data "m=" section as an "a=sctp-init" line.

5.3. Answerer Processing of the SDP Offer

If the answering endpoint negotiates a data "m=" section, it will parse the "a=sctp-init" line from the data "m=" section, if present.

If the data is not properly base64-encoded this results in an error.

The answering endpoint then informs its SCTP implementation of the series of bytes. The SCTP implementation is responsible for validating the format of the SCTP INIT chunk.

If the series of bytes is not a valid SCTP INIT chunk this results in an error.

5.4. Generating the SDP Answer

An endpoint not supporting the extension described in this document MUST NOT include a "a=sctp-init" attribute in its answer.

If the answering endpoint negotiated a data "m=" section and detected a valid "a=sctp-init" line in the offer, as described above, it MUST let its SCTP implementation generate a serialized INIT chunk that it would normally send over the network as a series of bytes.

This message is base64-encoded and added to the data "m=" section as a "a=sctp-init" line.

5.5. Offerer Processing of the SDP Answer

If the answer negotiated a data "m=" section, the offering endpoint parses the "a=sctp-init" line, if present, from the data "m=" section.

If the data is not properly base64-encoded this results in an error.

The offering endpoint then informs its SCTP implementation of the series of bytes. The SCTP implementation is responsible for validating the format of the SCTP INIT chunk. If the series of bytes is not a valid SCTP INIT chunk this results in an error.

5.6. Modifying the Session

Subsequent offers and answers MUST include the "a=sctp-init" line in the negotiated SDP with the same value as in the initial negotiation.

Remote offers MAY negotiate a new "a=sctp-init" line in conjunction with either * a new SCTP association as described in Section 9.3 of [RFC8841] * or a new DTLS association as described in Section 5.5 of [RFC8842].

Attempting to add an sctp-init attribute to an existing SCTP association results in an error.

6. SCTP considerations

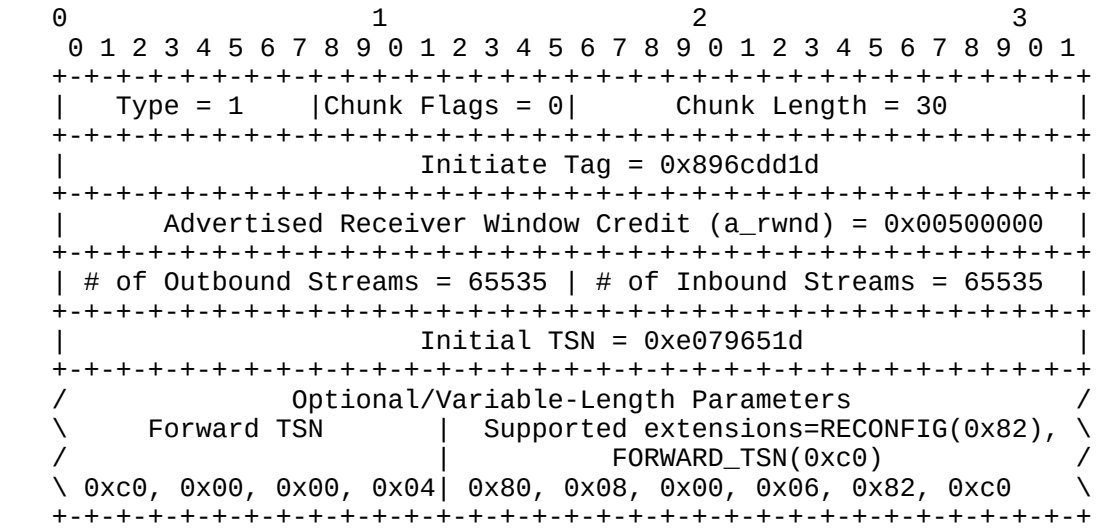
The creation of the "sctp-init" attribute SHOULD NOT change the state of the SCTP association into the COOKIE-WAIT state as described in Section 5.1 of [RFC9260] and SHOULD NOT start the T1-init timer.

Processing of the "sctp-init" attribute from the remote side SHOULD NOT change the state of the SCTP association and SHOULD NOT start any timer.

When the "sctp-init" attribute has been negotiated, both endpoints SHOULD consider the SCTP association to be in the ESTABLISHED state, as described in Section 4 of [RFC9260], once the DTLS handshake finishes. The steps A-E described in Section 5.1 of [RFC9260] can be skipped.

7. Example negotiation

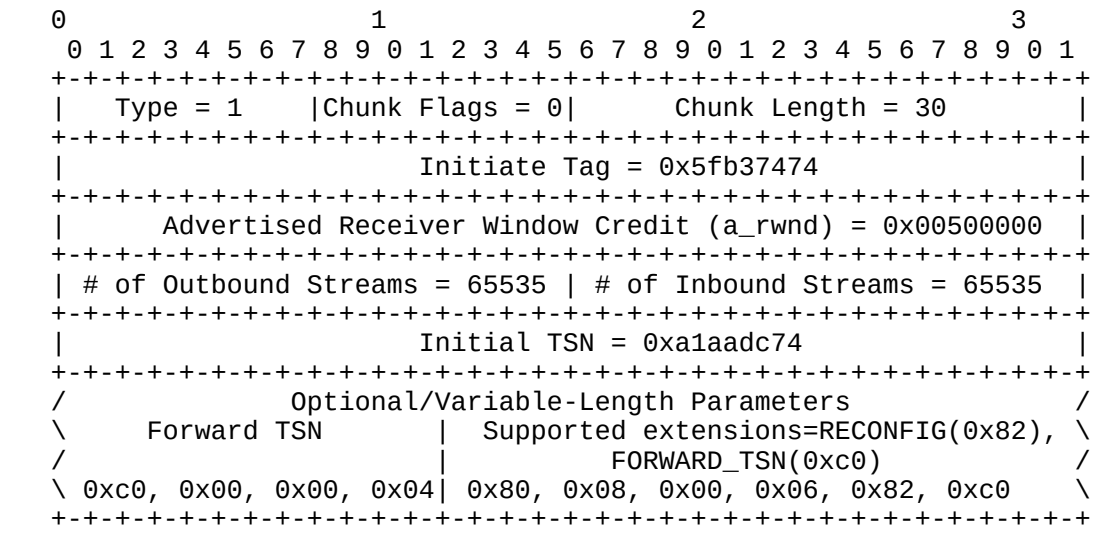
The offering client generates a SCTP INIT chunk with its supported parameters:



This is encoded in the offer SDP as follows:

```
v=0
o=- 8389853828849686268 2 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0
m=application 9 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
a=ice-ufrag:UgEn
a=ice-pwd:f/+ugRILrIUlAkSmkStnZb/h
a=ice-options:trickle
a=fingerprint:sha-256
              6A:15:F0:08:9C:55:51:CD:55:27:BD:0D:FB:14:DD:41:
              F6:8C:82:9F:CA:AD:DA:E7:04:61:6F:A9:FF:99:2D:7A
a=setup:actpass
a=mid:0
a=sctp-port:5000
a=max-message-size:262144
a=sctp-init:AQAAHols3R0AUAAA/////+B5ZR3AAAAEgAgABoLA
```

In response, the answering client generates its SCTP INIT chunk with its supported parameters:



This yields an answer SDP shown below:

```
v=0
o=- 6156433258406980035 2 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0
m=application 9 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
a=ice-ufrag:thzM
a=ice-pwd:YT278C3KYFmsedR5+OzE9OQY
a=ice-options:trickle
a=fingerprint:sha-256
              AC:48:A1:C4:78:6D:6D:46:63:BB:7D:70:E4:B8:5D:C4:
              E8:6E:66:40:8C:81:31:D3:9C:34:32:9A:C3:6B:BB:AE
a=setup:active
a=mid:0
a=sctp-port:5000
a=max-message-size:262144
a=sctp-init:AQAAHl+zdHQAUAAA/////6Gq3HTAAAAEgAgABoLA
```

8. Security Considerations

SNAP removes SCTP’s anti-amplification mechanism in order to accelerate connection startup. However, when SCTP runs atop DTLS as specified in [RFC8261], any attempt to send junk traffic over SCTP will fail as it will not be properly encrypted. Therefore SNAP does not add any new amplification risk.

Exposing the content of the SCTP INIT chunk, in particular the "Initiate Tag", in the SDP does not introduce new security concerns since running SCTP atop DTLS protects against the off-path attacks described in Section 5.3.1 of [RFC9260].

Exposing the content of the SCTP init tag to the application layer, e.g., JavaScript applications or the signaling channel in the case of WebRTC does allow these to add or remove variable-length parameters. Since these parameters are optional and used for feature negotiation, removing one is equivalent to the remote side not supporting the parameter and does not introduce a new security risk. Adding a parameter that is not supported may cause the remote side to send those but this is equivalent to receiving a packet for a feature that was not negotiated and hence does not introduce a new security concern.

9. IANA Considerations

9.1. New SDP attributes

This document defines a new SDP media-level attribute, "sctp-init". The details of the attribute are defined in Section 4.2.

10. References

10.1. 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/rfc/rfc2119>.

[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/rfc/rfc8174>.

[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/rfc/rfc9260>.

10.2. Informative References

[RFC8261] Tuexen, M., Stewart, R., Jesup, R., and S. Loreto, "Datagram Transport Layer Security (DTLS) Encapsulation of SCTP Packets", RFC 8261, DOI 10.17487/RFC8261, November 2017, <https://www.rfc-editor.org/rfc/rfc8261>.

[RFC8831] Jesup, R., Loreto, S., and M. T端xen, "WebRTC Data Channels", RFC 8831, DOI 10.17487/RFC8831, January 2021, <https://www.rfc-editor.org/rfc/rfc8831>.

[RFC8832] Jesup, R., Loreto, S., and M. T端xen, "WebRTC Data Channel Establishment Protocol", RFC 8832, DOI 10.17487/RFC8832, January 2021, <https://www.rfc-editor.org/rfc/rfc8832>.

[RFC8841] Holmberg, C., Shpount, R., Loreto, S., and G. Camarillo, "Session Description Protocol (SDP) Offer/Answer Procedures for Stream Control Transmission Protocol (SCTP) over Datagram Transport Layer Security (DTLS) Transport", RFC 8841, DOI 10.17487/RFC8841, January 2021, <https://www.rfc-editor.org/rfc/rfc8841>.

[RFC8842] Holmberg, C. and R. Shpount, "Session Description Protocol (SDP) Offer/Answer Considerations for Datagram Transport Layer Security (DTLS) and Transport Layer Security (TLS)", RFC 8842, DOI 10.17487/RFC8842, January 2021, <https://www.rfc-editor.org/rfc/rfc8842>.

Acknowledgments

The authors wish to thank Harald Alvestrand, Lennart Grahl and Jonas Oreland for their invaluable comments.

Authors' Addresses

Philipp Hancke
Meta Platforms Inc.
Email: philipp.hancke@googlemail.com

Justin Uberti
OpenAI
Email: justin@uberti.name

Victor Boivie
Google
Email: boivie@google.com