

RTCWeb Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 30, 2013

R. Jesup
Mozilla
S. Loreto
Ericsson
M. Tuexen
Muenster Univ. of Appl. Sciences
February 26, 2013

WebRTC Data Channel Protocol
draft-jesup-rtcweb-data-protocol-04.txt

Abstract

The Web Real-Time Communication (WebRTC) working group is charged to provide protocols to support for direct interactive rich communication using audio, video, and data between two peers' web-browsers. This document specifies an actual (minor) protocol for how the JS-layer DataChannel objects provide the data channels between the peers.

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 <http://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 August 30, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Conventions	3
3. Terminology	3
4. Protocol Overview	3
5. Opening Handshake	4
6. Control Messages	4
6.1. DATA_CHANNEL_OPEN Message	4
7. Procedures	6
7.1. Adding a Channel	6
7.2. Closing a Channel	7
7.3. Sending and Receiving Data	7
8. Security Considerations	7
9. IANA Considerations	7
10. Acknowledgments	8
11. References	8
11.1. Normative References	8
11.2. Informational References	9
Authors' Addresses	9

1. Introduction

The DataChannel Protocol is designed to provide, in the WebRTC context [I-D.ietf-rtcweb-overview], a generic transport service allowing Web Browser to exchange generic data in a bidirectional peer to peer fashion. As discussed in [I-D.ietf-rtcweb-data-channel] the protocol uses Stream Control Transmission Protocol (SCTP) [RFC4960] encapsulated on Datagram Transport Layer Security (DTLS) [RFC6347] as described in [I-D.tuexen-tsvwg-sctp-dtls-encaps] to benefit from their already standardized transport and security features.

2. Conventions

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 [RFC2119].

3. Terminology

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.

Channel: Two Streams with the same identifier, one in each direction, that are managed together.

4. Protocol Overview

This protocol is a simple, low-overhead way to establish bidirectional Channels over an SCTP association with a consistent set of properties.

Channels are created by sending an DATA_CHANNEL_OPEN message on an unused Stream. There is no handshake, and the channel is available to send on as soon as the DATA_CHANNEL_OPEN has been sent.

To avoid glare in opening Channels, each side must use either even or odd Streams when sending a DATA_CHANNEL_OPEN message. The method used to determine which side uses odd or even is TBD and may be based on DTLS connection roles when used in rtcweb.

There is no attempt to resolve label glare; if both sides open a Channel labelled "x" at the same time, there will be two Channels labelled "x" - one on an even Stream pair, one on an odd pair.

The protocol field is to ease cross-application interoperability ("federation") by identifying the data being passed with an IANA-registered string.

Data that arrives which on an unused Stream MUST be held until a DATA_CHANNEL_OPEN arrives for that Channel, or if the protocol stack had been told to expect data on that Stream and deliver it immediately, or until [TBD - report error]. This allows for external negotiation of streams (or assumption of negotiation by cooperating applications). If a later DATA_CHANNEL_OPEN arrives that conflicts with the pre-set properties of the Channel, an error should be signaled to higher levels.

Channels are closed by resetting the Stream.

5. Opening Handshake

The opening handshake is based on the multimedia session description exchange that happens between the browsers, typically through a Web Server acting as the signaling service.

[I-D.ietf-mmusic-sctp-sdp] defines the protocol identifier, 'SCTP/DTLS', and defines how to establish an SCTP association over DTLS using the Session Description Protocol (SDP).

The SCTP association is created with the number of streams specified by the application, and if not specified, then it SHOULD default to 16 streams.

It is recommended that additional streams be available dynamically based on [RFC6525].

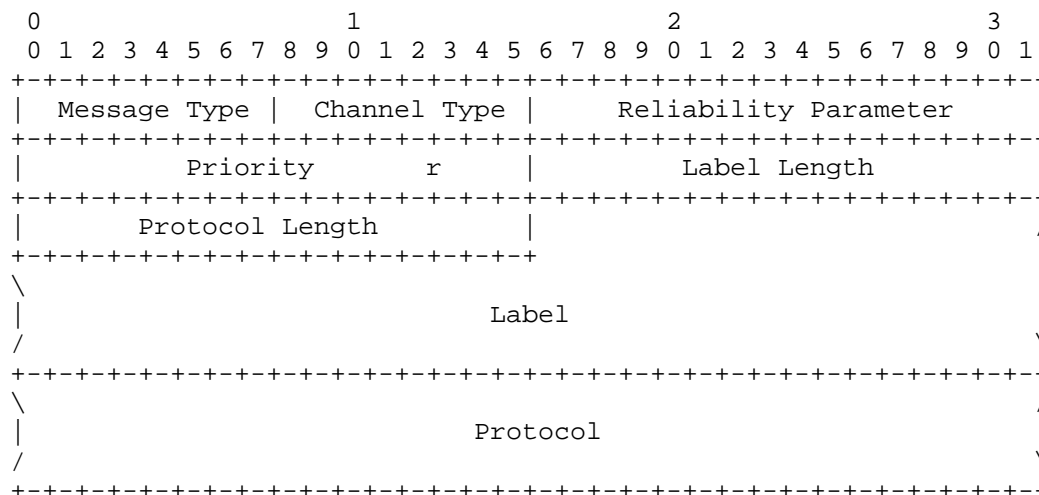
6. Control Messages

Control Messages are sent to manage opening bidirectional channels.

A DATA_CHANNEL_OPEN message is sent on the Stream that is intended to be used to send in that direction, and this creates a bidirectional Channel that may be used by both sides to send data.

6.1. DATA_CHANNEL_OPEN Message

This message is sent initially on the stream used for user messages using the channel. All DATA_CHANNEL_OPEN messages MUST be sent reliably and in-order.



Message Type: 1 byte (unsigned integer)

This field holds the IANA defined message type for the the DATA_CHANNEL_OPEN message. The suggested value of this field for IANA is 0x03. NOTE: values 0x00-0x02 were used in an older draft with incompatible structures. Any future incompatible message changes should define new message types.

Channel Type: 1 byte (unsigned integer)

This field specifies the type of the channel to be opened:

DATA_CHANNEL_RELIABLE (0x00): The channel provides a reliable in-order bi-directional communication channel.

DATA_CHANNEL_RELIABLE_UNORDERED (0x80): The channel provides a reliable unordered bi-directional communication channel.

DATA_CHANNEL_PARTIAL_RELIABLE_REXMIT (0x01): The channel provides a partially-reliable in-order bi-directional Communication channel. User messages will not be retransmitted more times than specified in the Reliability Parameter.

DATA_CHANNEL_PARTIAL_RELIABLE_REXMIT_UNORDERED (0x81): The channel provides a partial reliable unordered bi-directional Communication channel. User messages will not be retransmitted more times than specified in the Reliability Parameter.

DATA_CHANNEL_PARTIAL_RELIABLE_TIMED (0x02): The channel provides a partial reliable in-order bi-directional Communication channel. User messages might not be transmitted or retransmitted after a specified life-time given in milliseconds in the Reliability Parameter. This life-time starts when providing the user message to the Javascript engine.

DATA_CHANNEL_PARTIAL_RELIABLE_TIMED (0x82): The channel provides a partial reliable unordered bi-directional Communication channel. User messages might not be transmitted or retransmitted after a specified life-time given in milliseconds in the Reliability Parameter. This life-time starts when providing the user message to the Javascript engine.

Reliability Parameter: 2 bytes (unsigned integer)
This field is ignored if a reliable channel is used.
If a partial reliable channel with limited number of retransmissions is used, this field specifies the number of retransmissions. If a partial reliable channel with limited lifetime is used, this field specifies the maximum lifetime in milliseconds.

Priority: 2 bytes (integer)
The priority of the channel.

Label Length: 2 bytes (integer)
The length of the label field in bytes.

Protocol Length: 2 bytes (integer)
The length of the protocol field in bytes.

Label: Variable Length (sequence of characters)
The name of the channel. This may be an empty string.

Protocol: Variable Length (sequence of characters)
The protocol for the channel. This may be an empty string. If used, it SHOULD be an IANA-registered protocol.

7. Procedures

7.1. Adding a Channel

When one side wants to add a channel, it picks an unused outgoing stream (either even or odd, depending on TBD); if no unused streams are available a negotiation to increase the number is done. It should also check that the other side has the same channel available, and if not then initiate an increase in the number of streams. It then sends a DATA_CHANNEL_OPEN control message on the outgoing stream.

When an DATA_CHANNEL_OPEN is received on an incoming stream, the Stream is associated with the newly-created Channel. If any data had arrived on the Stream before the Open arrives and had been buffered, it is now released on the new Channel.

The channel_type and reliability_parameters fields of the DATA_CHANNEL_OPEN message MUST be used to set up the reverse side of the Channel so that both directions use the same options by default.

7.2. Closing a Channel

Channels MUST be closed by resetting the outgoing stream. If an incoming stream is reset by the peer, an corresponding outgoing stream reset SHOULD be issued. If both streams of a channel are reset, the channel is closed and the streams are available for reuse for new channel opens.

7.3. Sending and Receiving Data

Data shall be sent using PPID's other than the Data Channel Control PPID. These PPID's should be registered with IANA via (TBD). The meaning of these data PPIDs and the format of the data shall be specific to the usage of this protocol, and typically shall be provided to the higher layers to allow proper decoding of the data.

It is RECOMMENDED that higher layers wishing to transfer large messages fragment them using PPIDs or other mechanisms to avoid monopolization of the SCTP association by the transfer of a single large message, unless a future SCTP draft relaxes this concern. If fragmented solely with PPID values, then transmission must occur on a reliable in-order channel. If in-band application framing is used, then other options may be possible.

For WebRTC, data PPID's for DOMStrings and binary data (and fragmentation thereof) shall be created.

All data sent on a Channel in both directions MUST be sent over the underlying Stream using the reliability defined when the Channel was opened unless the options are changed, or per-message options are specified by a higher level.

Data may be sent immediately after sending or receiving a DATA_CHANNEL_OPEN message.

It is recommended that message size be kept within certain size bounds (TBD) as applications will not be able to support arbitrarily-large single messages.

8. Security Considerations

To be done.

9. IANA Considerations

This document also defines three new SCTP Payload Protocol

Identifiers (PPIDs). RFC 4960 [RFC4960] creates the registry from which these identifiers have been assigned. The following values have been reserved:

WebRTC Control - #To Be Assigned

NOTE: not needed if Stream 0 is dedicated to control

DOMString - #To Be Assigned

Binary Data Partial - #To Be Assigned

Binary Data Last - #To Be Assigned

10. Acknowledgments

The authors wish to thank Martin Thompson, Cullen Jennings, Harald Alvestrand, Peter Thatcher, Adam Bergkvist, Justin Uberti, Randall Stewart, Stefan Haekansson and many others for their invaluable comments.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, May 2004.
- [RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6525] Stewart, R., Tuexen, M., and P. Lei, "Stream Control Transmission Protocol (SCTP) Stream Reconfiguration", RFC 6525, February 2012.
- [I-D.ietf-mmusic-sctp-sdp] Loreto, S. and G. Camarillo, "Stream Control Transmission Protocol (SCTP)-Based Media Transport in the Session Description Protocol (SDP)", draft-ietf-mmusic-sctp-sdp-03 (work in progress), January 2013.
- [I-D.tuexen-tsvwg-sctp-dtls-encaps] Jesup, R., Loreto, S., Stewart, R., and M. Tuexen, "DTLS

Encapsulation of SCTP Packets for RTCWEB",
draft-tuexen-tsvwg-sctp-dtls-encaps-01 (work in progress),
July 2012.

11.2. Informational References

[I-D.ietf-rtcweb-overview]

Alvestrand, H., "Overview: Real Time Protocols for Browser-based Applications", draft-ietf-rtcweb-overview-06 (work in progress), February 2013.

[I-D.ietf-rtcweb-data-channel]

Jesup, R., Loreto, S., and M. Tuexen, "RTCWeb Data Channels", draft-ietf-rtcweb-data-channel-03 (work in progress), February 2013.

Authors' Addresses

Randell Jesup
Mozilla
US

Email: randell-ietf@jesup.org

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
FI

Email: salvatore.loreto@ericsson.com

Michael Tuexen
Muenster University of Applied Sciences
Stegerwaldstrasse 39
Steinfurt 48565
DE

Email: tuexen@fh-muenster.de

