Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 31, 2015

H. Alvestrand
Google

February 27, 2015

Transports for WebRTC
draft-ietf-rtcweb-transports-08

Abstract

This document describes the data transport protocols used by WebRTC, including the protocols used for interaction with intermediate boxes such as firewalls, relays and NAT boxes.

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 31, 2015.

Copyright Notice

Copyright (c) 2015 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 2
2. Requirements language 3
3. Transport and Middlebox specification 3
  3.1. System-provided interfaces 3
  3.2. Ability to use IPv4 and IPv6 3
  3.3. Usage of temporary IPv6 addresses 4
  3.4. Middle box related functions 4
  3.5. Transport protocols implemented 5
4. Media Prioritization 6
  4.1. Usage of Quality of Service - DSCP and Multiplexing . . . 6
  4.2. Local prioritization 8
5. IANA Considerations 9
6. Security Considerations 9
7. Acknowledgements 9
8. References 9
  8.1. Normative References 9
  8.2. Informative References 11
Appendix A. Change log 12
  A.1. Changes from -00 to -01 12
  A.2. Changes from -01 to -02 12
  A.3. Changes from -02 to -03 13
  A.4. Changes from -03 to -04 13
  A.5. Changes from -04 to -05 13
  A.6. Changes from -05 to -06 14
  A.7. Changes from -06 to -07 14
  A.8. Changes from -07 to -08 14
Author's Address 14

1. Introduction

WebRTC is a protocol suite aimed at real time multimedia exchange between browsers, and between browsers and other entities.

WebRTC is described in the WebRTC overview document, [I-D.ietf-rtcweb-overview], which also defines terminology used in this document.

This document focuses on the data transport protocols that are used by conforming implementations, including the protocols used for interaction with intermediate boxes such as firewalls, relays and NAT boxes.

This protocol suite intends to satisfy the security considerations described in the WebRTC security documents, [I-D.ietf-rtcweb-security] and [I-D.ietf-rtcweb-security-arch].

This document describes requirements that apply to all WebRTC devices. When there are requirements that apply only to WebRTC browsers, this is called out by using the word "browser".

2. 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].

3. Transport and Middlebox specification

3.1. System-provided interfaces

The protocol specifications used here assume that the following protocols are available to the implementations of the WebRTC protocols:

- UDP. This is the protocol assumed by most protocol elements described.

- TCP. This is used for HTTP/WebSockets, as well as for TURN/SSL and ICE-TCP.

For both protocols, IPv4 and IPv6 support is assumed.

For UDP, this specification assumes the ability to set the DSCP code point of the sockets opened on a per-packet basis, in order to achieve the prioritizations described in [I-D.ietf-tsvwg-rtcweb-qos] (see Section 4.1) when multiple media types are multiplexed. It does not assume that the DSCP codepoints will be honored, and does assume that they may be zeroed or changed, since this is a local configuration issue.

Platforms that do not give access to these interfaces will not be able to support a conforming WebRTC implementation.

This specification does not assume that the implementation will have access to ICMP or raw IP.

3.2. Ability to use IPv4 and IPv6

Web applications running in a WebRTC browser MUST be able to utilize both IPv4 and IPv6 where available - that is, when two peers have only IPv4 connectivity to each other, or they have only IPv6 connectivity to each other, applications running in the WebRTC browser MUST be able to communicate.

When TURN is used, and the TURN server has IPv4 or IPv6 connectivity to the peer or its TURN server, candidates of the appropriate types MUST be supported. The "Happy Eyeballs" specification for ICE [I-D.martinsen-mmusic-ice-dualstack-fairness] SHOULD be supported.

3.3. Usage of temporary IPv6 addresses

The IPv6 default address selection specification [RFC6724] specifies that temporary addresses [RFC4941] are to be preferred over permanent addresses. This is a change from the rules specified by [RFC3484]. For applications that select a single address, this is usually done by the IPV6_PREFER_SRC_TMP preference flag specified in [RFC5014]. However, this rule is not completely obvious in the ICE scope. This is therefore clarified as follows:

When a client gathers all IPv6 addresses on a host, and both temporary addresses and permanent addresses of the same scope are present, the client SHOULD discard the permanent addresses before forming pairs. This is consistent with the default policy described in [RFC6724].

3.4. Middle box related functions

The primary mechanism to deal with middle boxes is ICE, which is an appropriate way to deal with NAT boxes and firewalls that accept traffic from the inside, but only from the outside if it is in response to inside traffic (simple stateful firewalls).

ICE [RFC5245] MUST be supported. The implementation MUST be a full ICE implementation, not ICE-Lite. A full ICE implementation allows interworking with both ICE and ICE-Lite implementations when they are deployed appropriately.

In order to deal with situations where both parties are behind NATs of the type that perform endpoint-dependent mapping (as defined in [RFC5128] section 2.4), TURN [RFC5766] MUST be supported.

WebRTC browsers MUST support configuration of STUN and TURN servers, both from browser configuration and from an application.

In order to deal with firewalls that block all UDP traffic, the mode of TURN that uses TCP between the client and the server MUST be supported, and the mode of TURN that uses TLS over TCP between the client and the server MUST be supported. See [RFC5766] section 2.1 for details.

In order to deal with situations where one party is on an IPv4 network and the other party is on an IPv6 network, TURN extensions for IPv6 [RFC6156] MUST be supported.

TURN TCP candidates, where the connection from the client's TURN server to the peer is a TCP connection, [RFC6062] MAY be supported.

However, such candidates are not seen as providing any significant benefit, for the following reasons.

First, use of TURN TCP candidates would only be relevant in cases which both peers are required to use TCP to establish a PeerConnection.

Second, that use case is supported in a different way by both sides establishing UDP relay candidates using TURN over TCP to connect to their respective relay servers.

Third, using TCP only between the endpoint and its relay may result in less issues with TCP in regards to real-time constraints, e.g. due to head of line blocking.

ICE-TCP candidates [RFC6544] MUST be supported; this may allow applications to communicate to peers with public IP addresses across UDP-blocking firewalls without using a TURN server.

If TCP connections are used, RTP framing according to [RFC4571] MUST be used, both for the RTP packets and for the DTLS packets used to carry data channels.

The ALTERNATE-SERVER mechanism specified in [RFC5389] (STUN) section 11 (300 Try Alternate) MUST be supported.

The WebRTC implementation MAY support accessing the Internet through an HTTP proxy. If it does so, it MUST support the "connect" header as specified in [I-D.ietf-httpbis-tunnel-protocol].

3.5. Transport protocols implemented

For transport of media, secure RTP is used. The details of the profile of RTP used are described in "RTP Usage" [I-D.ietf-rtcweb-rtp-usage]. Key exchange MUST be done using DTLS-SRTP, as described in [I-D.ietf-rtcweb-security-arch].

For data transport over the WebRTC data channel [I-D.ietf-rtcweb-data-channel], WebRTC implementations MUST support SCTP over DTLS over ICE. This encapsulation is specified in [I-D.ietf-tsvwg-sctp-dtls-encaps]. Negotiation of this transport in

SDP is defined in [I-D.ietf-mmusic-sctp-sdp]. The SCTP extension for NDATA, [I-D.ietf-tsvwg-sctp-ndata], MUST be supported.

The setup protocol for WebRTC data channels is described in [I-D.ietf-rtcweb-data-protocol].

WebRTC implementations MUST support multiplexing of DTLS and RTP over the same port pair, as described in the DTLS_SRTP specification [RFC5764], section 5.1.2. All application layer protocol payloads over this DTLS connection are SCTP packets.

Protocol identification MUST be supplied as part of the DTLS handshake, as specified in [I-D.ietf-rtcweb-alpn].

4. Media Prioritization

The WebRTC prioritization model is that the application tells the WebRTC implementation about the priority of media and data flows through an API.

The priority associated with a media or data flow is classified as "normal", "below normal", "high" or "very high". There are only four priority levels at the API.

The priority settings affect two pieces of behavior: Packet markings and packet send sequence decisions. Each is described in its own section below.

4.1. Usage of Quality of Service - DSCP and Multiplexing

Implementations SHOULD attempt to set QoS on the packets sent, according to the guidelines in [I-D.ietf-tsvwg-rtcweb-qos]. It is appropriate to depart from this recommendation when running on platforms where QoS marking is not implemented.

The implementation MAY turn off use of DSCP markings if it detects symptoms of unexpected behaviour like priority inversion or blocking of packets with certain DSCP markings. The detection of these conditions is implementation dependent. (Question: Does there need to be an API knob to turn off DSCP markings?)

All packets arrying data from the SCTP association supporting the data channels MUST use a single DSCP code point.

All packets on one TCP connection, no matter what it carries, MUST use a single DSCP code point.

More advice on the use of DSCP code points with RTP is given in [I-D.ietf-dart-dscp-rtp].

There exist a number of schemes for achieving quality of service that do not depend solely on DSCP code points. Some of these schemes depend on classifying the traffic into flows based on 5-tuple (source address, source port, protocol, destination address, destination port) or 6-tuple (5-tuple + DSCP code point). Under differing conditions, it may therefore make sense for a sending application to choose any of the configurations:

- Each media stream carried on its own 5-tuple
- Media streams grouped by media type into 5-tuples (such as carrying all audio on one 5-tuple)
- All media sent over a single 5-tuple, with or without differentiation into 6-tuples based on DSCP code points

In each of the configurations mentioned, data channels may be carried in its own 5-tuple, or multiplexed together with one of the media flows.

More complex configurations, such as sending a high priority video stream on one 5-tuple and sending all other video streams multiplexed together over another 5-tuple, can also be envisioned. More information on mapping media flows to 5-tuples can be found in [I-D.ietf-rtcweb-rtp-usage].

A sending implementation MUST be able to support the following configurations:

- multiplex all media and data on a single 5-tuple (fully bundled)
- send each media stream on its own 5-tuple and data on its own 5-tuple (fully unbundled)

It MAY choose to support other configurations, such as bundling each media type (audio, video or data) into its own 5-tuple (bundling by media type).

Sending data over multiple 5-tuples is not supported.

A receiving implementation MUST be able to receive media and data in all these configurations.

4.2. Local prioritization

When an WebRTC implementation has packets to send on multiple streams (with each media stream and each data channel considered as one "stream" for this purpose) that are congestion-controlled under the same congestion controller, the WebRTC implementation SHOULD cause data to be emitted in such a way that each stream at each level of priority is being given approximately twice the transmission capacity (measured in payload bytes) of the level below.

Thus, when congestion occurs, a "very high" priority flow will have the ability to send 8 times as much data as a "below normal" flow if both have data to send. This prioritization is independent of the media type. The details of which packet to send first are implementation defined.

For example: If there is a very high priority audio flow sending 100 byte packets, and a normal priority video flow sending 1000 byte packets, and outgoing capacity exists for sending >5000 payload bytes, it would be appropriate to send 4000 bytes (40 packets) of audio and 1000 bytes (one packet) of video as the result of a single pass of sending decisions.

Conversely, if the audio flow is marked normal priority and the video flow is marked very high priority, the scheduler may decide to send 2 video packets (2000 bytes) and 5 audio packets (500 bytes) when outgoing capacity exists for sending > 2500 payload bytes.

If there are two very high priority audio flows, each will be able to send 4000 bytes in the same period where a normal priority video flow is able to send 1000 bytes.

Two example implementation strategies are:

- When the available bandwidth is known from the congestion control algorithm, configure each codec and each data channel with a target send rate that is appropriate to its share of the available bandwidth.

- When congestion control indicates that a specified number of packets can be sent, send packets that are available to send using a weighted round robin scheme across the connections.

Any combination of these, or other schemes that have the same effect, is valid, as long as the distribution of transmission capacity is approximately correct.

For media, it is usually inappropriate to use deep queues for sending; it is more useful to, for instance, skip intermediate frames that have no dependencies on them in order to achieve a lower bitrate. For reliable data, queues are useful.

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. Security Considerations

Security considerations are enumerated in [I-D.ietf-rtcweb-security].

7. Acknowledgements

This document is based on earlier versions embedded in [I-D.ietf-rtcweb-overview], which were the results of contributions from many RTCWEB WG members.

Special thanks for reviews of earlier versions of this draft go to Eduardo Gueiros, Magnus Westerlund, Markus Isomaki and Dan Wing; the contributions from Andrew Hutton also deserve special mention.

8. References

8.1. Normative References

[I-D.ietf-httpbis-tunnel-protocol]
Hutton, A., Uberti, J., and M. Thomson, "The Tunnel-Protocol HTTP Request Header Field", draft-ietf-httpbis-tunnel-protocol-01 (work in progress), January 2015.

[I-D.ietf-mmusic-sctp-sdp]
Holmberg, C., Loreto, S., and G. Camarillo, "Stream Control Transmission Protocol (SCTP)-Based Media Transport in the Session Description Protocol (SDP)", draft-ietf-mmusic-sctp-sdp-12 (work in progress), January 2015.

[I-D.ietf-rtcweb-alpn]
Thomson, M., "Application Layer Protocol Negotiation for Web Real-Time Communications (WebRTC)", draft-ietf-rtcweb-alpn-00 (work in progress), July 2014.

[I-D.ietf-rtcweb-data-channel]
Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data Channels", draft-ietf-rtcweb-data-channel-13 (work in progress), January 2015.

[I-D.ietf-rtcweb-rtp-usage]
Perkins, C., Westerlund, M., and J. Ott, "Web Real-Time Communication (WebRTC): Media Transport and Use of RTP", draft-ietf-rtcweb-rtp-usage-22 (work in progress), February 2015.

[I-D.ietf-rtcweb-security]
Rescorla, E., "Security Considerations for WebRTC", draft-ietf-rtcweb-security-07 (work in progress), July 2014.

[I-D.ietf-rtcweb-security-arch]
Rescorla, E., "WebRTC Security Architecture", draft-ietf-rtcweb-security-arch-10 (work in progress), July 2014.

[I-D.ietf-tsvwg-rtcweb-qos]
Dhesikan, S., Jennings, C., Druta, D., Jones, P., and J. Polk, "DSCP and other packet markings for RTCWeb QoS", draft-ietf-tsvwg-rtcweb-qos-03 (work in progress), November 2014.

[I-D.ietf-tsvwg-sctp-dtls-encaps]
Tuexen, M., Stewart, R., Jesup, R., and S. Loreto, "DTLS Encapsulation of SCTP Packets", draft-ietf-tsvwg-sctp-dtls-encaps-09 (work in progress), January 2015.

[I-D.ietf-tsvwg-sctp-ndata]
Stewart, R., Tuexen, M., Loreto, S., and R. Seggelmann, "Stream Schedulers and a New Data Chunk for the Stream Control Transmission Protocol", draft-ietf-tsvwg-sctp-ndata-02 (work in progress), January 2015.

[I-D.martinsen-mmusic-ice-dualstack-fairness]
Martinsen, P., Reddy, T., and P. Patil, "ICE IPv4/IPv6 Dual Stack Fairness", draft-martinsen-mmusic-ice-dualstack-fairness-02 (work in progress), February 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4571] Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, July 2006.

[RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, May 2010.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, April 2010.

[RFC6062] Perreault, S. and J. Rosenberg, "Traversal Using Relays around NAT (TURN) Extensions for TCP Allocations", RFC 6062, November 2010.

[RFC6156] Camarillo, G., Novo, O., and S. Perreault, "Traversal Using Relays around NAT (TURN) Extension for IPv6", RFC 6156, April 2011.

[RFC6544] Rosenberg, J., Keranen, A., Lowekamp, B., and A. Roach, "TCP Candidates with Interactive Connectivity Establishment (ICE)", RFC 6544, March 2012.

[RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

8.2. Informative References

[I-D.ietf-dart-dscp-rtp] Black, D. and P. Jones, "Differentiated Services (DiffServ) and Real-time Communication", draft-ietf-dart-dscp-rtp-10 (work in progress), November 2014.

[I-D.ietf-rtcweb-data-protocol] Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data Channel Establishment Protocol", draft-ietf-rtcweb-data-protocol-09 (work in progress), January 2015.

[I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Browser-based Applications", draft-ietf-rtcweb-overview-13 (work in progress), November 2014.

[RFC3484] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, February 2003.

[RFC5014] Nordmark, E., Chakrabarti, S., and J. Laganier, "IPv6 Socket API for Source Address Selection", RFC 5014, September 2007.

[RFC5128] Srisuresh, P., Ford, B., and D. Kegel, "State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)", RFC 5128, March 2008.

Appendix A. Change log

This section should be removed before publication as an RFC.

A.1. Changes from -00 to -01

- Clarified DSCP requirements, with reference to -qos-
- Clarified "symmetric NAT" -> "NATs which perform endpoint-dependent mapping"
- Made support of TURN over TCP mandatory
- Made support of TURN over TLS a MAY, and added open question
- Added an informative reference to -firewalls-
- Called out that we don't make requirements on HTTP proxy interaction (yet

A.2. Changes from -01 to -02

- Required support for 300 Alternate Server from STUN.
- Separated the ICE-TCP candidate requirement from the TURN-TCP requirement.
- Added new sections on using QoS functions, and on multiplexing considerations.
- Removed all mention of RTP profiles. Those are the business of the RTP usage draft, not this one.

- Required support for TURN IPv6 extensions.

- Removed reference to the TURN URI scheme, as it was unnecessary.

- Made an explicit statement that multiplexing (or not) is an application matter.

.

A.3. Changes from -02 to -03

- Added required support for draft-ietf-tsvwg-sctp-ndata

- Removed discussion of multiplexing, since this is present in rtp-usage.

- Added RFC 4571 reference for framing RTP packets over TCP.

- Downgraded TURN TCP candidates from SHOULD to MAY, and added more language discussing TCP usage.

- Added language on IPv6 temporary addresses.

- Added language describing multiplexing choices.

- Added a separate section detailing what it means when we say that an WebRTC implementation MUST support both IPv4 and IPv6.

A.4. Changes from -03 to -04

- Added a section on prioritization, moved the DSCP section into it, and added a section on local prioritization, giving a specific algorithm for interpreting "priority" in local prioritization.

- ICE-TCP candidates was changed from MAY to MUST, in recognition of the sense of the room at the London IETF.

A.5. Changes from -04 to -05

- Reworded introduction

- Removed all references to "WebRTC". It now uses only the term RTCWEB.

- Addressed a number of clarity / language comments

- Rewrote the prioritization to cover data channels and to describe multiple ways of prioritizing flows

- Made explicit reference to "MUST do DTLS-SRTP", and referred to security-arch for details

A.6. Changes from -05 to -06

- Changed all references to "RTCWEB" to "WebRTC", except one reference to the working group

- Added reference to the httpbis "connect" protocol (being adopted by HTTPBIS)

- Added reference to the ALPN header (being adopted by RTCWEB)

- Added reference to the DART RTP document

- Said explicitly that SCTP for data channels has a single DSCP codepoint

A.7. Changes from -06 to -07

- Updated references

- Removed reference to draft-hutton-rtcweb-nat-firewall-considerations

A.8. Changes from -07 to -08

- Updated references

- Deleted "bundle each media type (audio, video or data) into its own 5-tuple (bundling by media type)" from MUST support configuration, since JSEP does not have a means to negotiate this configuration

Author's Address

Harald Alvestrand
Google

Email: harald@alvestrand.no