Internet Engineering Task Force (IETF)
Request for Comments: 7478
Category: Informational
ISSN: 2070-1721

C. Holmberg
S. Hakansson
G. Eriksson
Ericsson

March 2015

Web Real-Time Communication Use Cases and Requirements

Abstract

This document describes web-based real-time communication use cases. Requirements on the browser functionality are derived from the use cases.

This document was developed in an initial phase of the work with rather minor updates at later stages. It has not really served as a tool in deciding features or scope for the WG's efforts so far. It is being published to record the early conclusions of the WG. It will not be used as a set of rigid guidelines that specifications and implementations will be held to in the future.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7478.

Copyright Notice

Table of Contents

1. Introduction

This document presents a few use cases of web applications that are executed in a browser and use real-time communication capabilities. In most of the use cases, all end-user clients are web applications, but there are some use cases where at least one of the end-user clients is of another type (e.g., a mobile phone or a SIP User Agent (UA)).

Based on the use cases, the document derives requirements related to browser functionality. These requirements are named "Fn", where n is an integer, and are listed in conjunction with the use cases. A summary is provided in Section 3.2.

This document was developed in an initial phase of the work with rather minor updates at later stages. It has not really served as a tool in deciding features or scope for the WG's efforts so far. It is proposed to be used in a later phase to evaluate the protocols and solutions developed by the WG.

This document also lists requirements related to the API to be used by web applications as an appendix. The reason is that the W3C WebRTC WG has decided to not develop its own use-case or requirement document, but instead will use this document. These requirements are named "An", where n is an integer, and are described in Appendix A.

This document was developed in an initial phase of the work with rather minor updates at later stages. It has not really served as a tool in deciding features or scope for the WG's efforts so far. It is being published to record the early conclusions of the WG. It will not be used as a set of rigid guidelines that specifications and implementations will be held to in the future.

2. Use Cases

2.1. Introduction

This section describes web-based real-time communication use cases, from which requirements are derived.

The following considerations are applicable to all use cases:

- Clients can be on IPv4-only
- Clients can be on IPv6-only
- Clients can be on dual-stack

- Clients can be connected to networks with different throughput capabilities
- Clients can be on variable-media-quality networks (wireless)
- Clients can be on congested networks
- Clients can be on firewalled networks with no UDP allowed
- Clients can be on networks with a NAT or IPv4-IPv6 translation devices using any type of Mapping and Filtering behaviors (as described in RFC 4787).

2.2. Common Requirements

The requirements retrieved from the Simple Video Communication Service use case (Section 2.3.1) by default apply to all other use cases and are considered common. For each use case, only the additional requirements are listed.

2.3. Browser-to-Browser Use Cases

2.3.1. Simple Video Communication Service

2.3.1.1. Description

Two or more users have loaded a video communication web application into their browsers, provided by the same service provider, and logged into the service it provides. The web service publishes information about user login status by pushing updates to the web application in the browsers. When one online user selects a peer online user, a 1:1 audiovisual communication session between the browsers of the two peers is initiated. The invited user might accept or reject the session.

During session establishment, a self view is displayed, and once the session has been established the video sent from the remote peer is displayed in addition to the self view. During the session, each user can:

- select to remove and reinsert the self-view as often as desired,
- change the sizes of his/her two video displays during the session, and
- pause the sending of media (audio, video, or both) and mute incoming media.

It is essential that media and data be encrypted, authenticated, and integrity protected on a per-IP-packet basis and that media and data packets failing the integrity check not be delivered to the application.

The application gives the users the opportunity to stop it from exposing the host IP address to the application of the other user.

Any session participant can end the session at any time.

The two users may be using communication devices with different operating systems and browsers from different vendors.

The web service monitors the quality of the service (focus on quality of audio and video) that the end users experience.

2.3.1.2. Common Requirements

REQ-ID	DESCRIPTION
F1	The browser must be able to use microphones and cameras as input devices to generate streams.
F2	The browser must be able to send streams and data to a peer in the presence of NATs.
F3	Transmitted streams and data must be rate controlled (meaning that the browser must, regardless of application behavior, reduce send rate when there is congestion).
F4	The browser must be able to receive, process, and render streams and data ("render" does not apply for data) from peers.
F5	The browser should be able to render good quality audio and video even in the presence of reasonable levels of jitter and packet losses.
F6	The browser must detect when a stream from a peer is not received anymore.

F7	When there are both incoming and outgoing audio streams, echo cancellation must be made available to avoid disturbing echo during conversation.
F8	The browser must support synchronization of audio and video.
F9	The browser should use encoding of streams suitable for the current rendering (e.g., video display size) and should change parameters if the rendering changes during the session.
F10	The browser must support a baseline audio and video codec.
F11	It must be possible to protect streams and data from wiretapping [RFC2804] [RFC7258].
F12	The browser must enable verification, given the right circumstances and by use of other trusted communication, that streams and data received have not been manipulated by any party.
F13	The browser must encrypt, authenticate, and integrity protect media and data on a per-IP-packet basis, and it must drop incoming media and data packets that fail the per-IP-packet integrity check. In addition, the browser must support a mechanism for cryptographically binding media and data security keys to the user identity (see R-ID-BINDING in [RFC5479]).
F14	The browser must make it possible to set up a call between two parties without one party learning the other party's host IP address.
F15	The browser must be able to collect statistics, related to the transport of audio and video between peers, needed to estimate quality of experience.

A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A25, A26

2.3.2. Simple Video Communication Service: NAT/Firewall That Blocks UDP

2.3.2.1. Description

This use case is almost identical to the Simple Video Communication Service use case (Section 2.3.1). The difference is that one of the users is behind a NAT/firewall that blocks UDP traffic.

2.3.2.2. Additional Requirements

REQ-ID	DESCRIPTION
F18	The browser must be able to send streams and data to a peer in the presence of NATs and firewalls that block UDP traffic.

2.3.3. Simple Video Communication Service: Firewall That Only Allows Traffic via an HTTP Proxy

2.3.3.1. Description

This use case is almost identical to the Simple Video Communication Service use case (Section 2.3.1). The difference is that one of the users is behind a firewall that only allows traffic via an HTTP Proxy.

2.3.3.2. Additional Requirements

REQ-ID	DESCRIPTION
F21	The browser must be able to send streams and data to a peer in the presence of firewalls that only allow traffic via an HTTP Proxy, when firewall policy allows WebRTC traffic.

2.3.4. Simple Video Communication Service: Global Service Provider

2.3.4.1. Description

This use case is almost identical to the Simple Video Communication Service use case (Section 2.3.1). What is added is that the service provider is operating over large geographical areas (or even globally).

Assuming that the Interactive Connectivity Establishment (ICE) mechanism [RFC5245] will be used, this means that the service provider would like to be able to provide several Session Traversal Utilities for NAT (STUN) and Traversal Using Relay NAT (TURN) servers (via the app) to the browser; selection of which one(s) to use is part of the ICE processing. Other reasons for wanting to provide several STUN and TURN servers include support for IPv4 and IPv6, load balancing, and redundancy.

Note that ICE support being mandatory does not preclude a WebRTC endpoint from supporting more traversal mechanisms than ICE using STUN and TURN.

2.3.4.2. Additional Requirements

REQ-ID	DESCRIPTION
F19	The browser must be able to use several STUN and TURN servers.

A22

2.3.5. Simple Video Communication Service: Enterprise Aspects

2.3.5.1. Description

This use case is similar to the Simple Video Communication Service use case (Section 2.3.1).

What is added is aspects when using the service in enterprises. ICE is assumed in the further description of this use case.

An enterprise that uses a WebRTC-based web application for communication desires to audit all WebRTC-based application sessions used from inside the company towards any external peer. To be able to do this, they deploy a TURN server that straddles the boundary between the internal and the external network.

The firewall will block all attempts to use STUN with an external destination unless they go to the enterprise auditing TURN server. In cases where employees are using WebRTC applications provided by an external service provider, they still want the traffic to stay inside their internal network and in addition not load the straddling TURN server; thus, they deploy a STUN server allowing the WebRTC client to determine its server reflexive address on the internal side. Thus, enabling cases where peers are both on the internal side to connect

without the traffic leaving the internal network. It must be possible to configure the browsers used in the enterprise with network specific STUN and TURN servers. This should be possible to achieve by autoconfiguration methods. The WebRTC functionality will need to utilize both network specific STUN and TURN resources and STUN and TURN servers provisioned by the web application.

2.3.5.2. Additional Requirements

REQ-ID	DESCRIPTION
F20	The browser must support the use of STUN and TURN servers that are supplied by entities other than the web application (i.e., the network provider).

2.3.6. Simple Video Communication Service: Access Change

2.3.6.1. Description

This use case is almost identical to the Simple Video Communication Service use case (Section 2.3.1). The difference is that the user changes network access during the session.

The communication device used by one of the users has several network adapters (Ethernet, Wi-Fi, Cellular). The communication device is accessing the Internet using Ethernet, but the user has to start a trip during the session. The communication device automatically changes to use Wi-Fi when the Ethernet cable is removed and then moves to cellular access to the Internet when moving out of Wi-Fi coverage. The session continues even though the access method changes.

2.3.6.2. Additional Requirements

REQ-ID	DESCRIPTION
F17	The communication session must survive across a change of the network interface used by the session.

2.3.7. Simple Video Communication Service: QoS

2.3.7.1. Description

This use case is almost identical to the Simple Video Communication Service: Access Change use case (Section 2.3.6). The use of Quality of Service (QoS) capabilities is added:

The user in the previous use case that starts a trip is behind a common residential router that supports differentiation of traffic. In addition, the user's provider of cellular access has QoS support enabled. The user is able to take advantage of the QoS support both when accessing via the residential router and when using cellular.

2.3.7.2. Additional Requirements

REQ-ID	DESCRIPTION
F17	The communication session must survive across a change of the network interface used by the session.
F22	The browser should be able to take advantage of available capabilities (supplied by network nodes) to differentiate voice, video, and data appropriately.

2.3.8. Simple Video Communication Service with Screen Sharing

2.3.8.1. Description

This use case has the audio and video communication of the Simple Video Communication Service use case (Section 2.3.1).

However, in addition to this, one of the users can share what is being displayed on her/his screen with a peer. The user can choose to share the entire screen, part of the screen (part selected by the user), or what a selected application displays with the peer.

2.3.8.2. Additional Requirements

REQ-ID	DESCRIPTION
F36	The browser must be able to generate streams using the entire user display, a specific area of the user display, or the information being displayed by a specific application.

A21

2.3.9. Simple Video Communication Service with File Exchange

2.3.9.1. Description

This use case has the audio and video communication of the Simple Video Communication Service use case (Section 3.3.1).

However, in addition to this, the users can send and receive files stored in the file system of the device used.

2.3.9.2. Additional Requirements

REQ-ID	DESCRIPTION
F35	The browser must be able to send reliable data traffic to a peer browser.

A21, A24

2.3.10. Hockey Game Viewer

2.3.10.1. Description

An ice-hockey club uses an application that enables talent scouts to, in real-time, show and discuss games and players with the club manager. The talent scouts use a mobile phone with two cameras: one front facing and one rear facing.

The club manager uses a desktop, equipped with one camera, for viewing the game and discussing with the talent scout.

Before the game starts, and during game breaks, the talent scout and the manager have a 1:1 audiovisual communication session. On the mobile phone, only the camera facing the talent scout is used. On the user display of the mobile phone, the video of the club manager is shown with a picture-in-picture thumbnail of the rear-facing camera (self view). On the display of the desktop, the video of the talent scout is shown with a picture-in-picture thumbnail of the desktop camera (self view).

When the game is ongoing, the talent scout activates the use of the front-facing camera, and that stream is sent to the desktop (the stream from the rear-facing camera continues to be sent all the time). The video stream captured by the front-facing camera (that is capturing the game) of the mobile phone is shown in a big window on the desktop screen, with picture-in-picture thumbnails of the rear-facing camera and the desktop camera (self view). On the display of the mobile phone the game is shown (front-facing camera) with picture-in-picture thumbnails of the rear-facing camera (self view) and the desktop camera. Because the most important stream in this phase is the video showing the game, the application used in the talent scout's mobile phone sets higher priority for that stream.

2.3.10.2. Additional Requirements

REQ-ID	DESCRIPTION
F22	The browser should be able to take advantage of available capabilities (supplied by network nodes) to differentiate voice, video, and data appropriately.
F25	The browser must be able to render several concurrent audio and video streams.

A17, A23

2.3.11. Multiparty Video Communication

2.3.11.1. Description

In this use case, the Simple Video Communication Service use case (Section 2.3.1) is extended by allowing multiparty sessions. No central server is involved -- the browser of each participant sends and receives streams to and from all other session participants. The web application in the browser of each user is responsible for setting up streams to all receivers.

In order to enhance the user experience, the web application renders the audio coming from different participants so that it is experienced to come from different spatial locations. This is done automatically, but users can change how the different participants are placed in the (virtual) room. In addition, the levels in the audio signals are adjusted before mixing.

Another feature intended to enhance the user experience is the highlighting of the video window that displays the video of the currently speaking peer.

Each video stream received is, by default, displayed in a thumbnail frame within the browser, but users can change the display size.

Note: What this use case adds in terms of requirements are capabilities to send streams to and receive streams from several peers concurrently as well as the capabilities to render the video from all received streams and be able to spatialize, level adjust, and mix the audio from all received streams locally in the browser. It also adds the capability to measure the audio level/activity.

2.3.11.2. Additional Requirements

REQ-ID	DESCRIPTION
F23	The browser must be able to transmit streams and data to several peers concurrently.
F24	The browser must be able to receive streams and data from multiple peers concurrently.
F25	The browser must be able to render several concurrent audio and video streams.
F26	The browser must be able to mix several audio streams.
F27	The browser must be able to apply spatialization effects to audio streams.
F28	The browser must be able to measure the voice activity level in audio streams.
F29	The browser must be able to change the voice activity level in audio streams.

A13, A14, A15, A16

2.3.12. Multiparty Online Game with Voice Communication

2.3.12.1. Description

This use case is based on the previous one. In this use case, the voice part of the multiparty video communication use case is used in the context of an online game. The received voice audio media is rendered together with game sound objects. For example, the sound of a tank moving from left to right over the screen must be rendered and played to the user together with the voice media.

Quick updates of the game state are required, and they have higher priority than the voice.

Note: the difference regarding local audio processing compared to the "Multiparty Video Communication" use case is that other sound objects than the streams must be possible to be included in the

spatialization and mixing. "Other sound objects" could for example be a file with the sound of the tank; that file could be stored locally or remotely.

2.3.12.2. Additional Requirements

REQ-ID	DESCRIPTION
F22	The browser should be able to take advantage of available capabilities (supplied by network nodes) to differentiate voice, video, and data appropriately.
F23	The browser must be able to transmit streams and data to several peers concurrently.
F24	The browser must be able to receive streams and data from multiple peers concurrently.
F25	The browser must be able to render several concurrent audio and video streams.
F26	The browser must be able to mix several audio streams.
F27	The browser must be able to apply spatialization effects when playing audio streams.
F28	The browser must be able to measure the voice activity level in audio streams.
F29	The browser must be able to change the voice activity level in audio streams.
F30	The browser must be able to process and mix sound objects (media that is retrieved from another source than the established media stream(s) with the peer(s) with audio streams).
F34	The browser must be able to send short latency unreliable datagram traffic to a peer browser [RFC5405].

A13, A14, A15, A16, A17, A18, A23

2.4. Browser - GW/Server Use Cases

2.4.1. Telephony Terminal

2.4.1.1. Description

A mobile telephony operator allows its customers to use a web browser to access their services. After a simple log in, the user can place and receive calls in the same way as when using a normal mobile phone. When a call is received or placed, the identity is shown in the same manner as when a mobile phone is used.

Note: "place and receive calls in the same way as when using a normal mobile phone" means that you can dial a number and your mobile telephony operator has made available your phone contacts online so that they are available and can be clicked to call and they can be used to present the identity of an incoming call. If the callee is not in your phone contacts, the number is displayed. Furthermore, your call logs are available, and updated with the calls made/ received from the browser. For people receiving calls made from the web browser, the usual identity (i.e., the phone number of the mobile phone) will be presented.

2.4.1.2. Additional Requirements

REQ-ID	DESCRIPTION
F31	The browser must support an audio media format (codec) that is commonly supported by existing telephony services.
F33	The browser must be able to initiate and accept a media session where the data needed for establishment can be carried in SIP.

2.4.2. FedEx Call

2.4.2.1. Description

Alice uses her web browser with a service that allows her to call Public Switched Telephone Network (PSTN) numbers. Alice calls 1-800-123-4567. Alice should be able to hear the initial prompts from the FedEx Interactive Voice Responder (IVR), and when the IVR says press 1, there should be a way for Alice to navigate the IVR.

2.4.2.2. Additional Requirements

REQ-ID	DESCRIPTION
F31	The browser must support an audio media format (codec) that is commonly supported by existing telephony services.
F32	There should be a way to navigate a dual-tone multi-frequency signaling (DTMF) based Interactive Voice Response (IVR) system.

2.4.3. Video Conferencing System with Central Server

2.4.3.1. Description

An organization uses a video communication system that supports the establishment of multiparty video sessions using a central conference server.

The browser of each participant sends an audio stream (type in terms of mono, stereo, 5.1 -- depending on the equipment of the participant) to the central server. The central server mixes the audio streams (and can in the mixing process naturally add effects such as spatialization) and sends towards each participant a mixed audio stream that is played to the user.

The browser of each participant sends video towards the server. For each participant, one high-resolution video is displayed in a large window, while a number of low-resolution videos are displayed in smaller windows. The server selects what video streams to be forwarded as main and thumbnail videos, respectively, based on speech activity. As the video streams to display can change quite frequently (as the conversation flows), it is important that the delay from when a video stream is selected for display until the video can be displayed is short.

All participants are authenticated by the central server and authorized to connect to the central server. The participants are identified to each other by the central server, and the participants do not have access to each others' credentials such as email addresses or login IDs.

Note: This use case adds requirements on support for fast stream switches (F16). There exist several solutions that enable the server to forward one high-resolution and several low-resolution video

streams: a) each browser could send a high-resolution, but scalable stream, and the server could send just the base layer for the low-resolution streams, b) each browser could in a simulcast fashion send one high-resolution and one low-resolution stream, and the server just selects, or c) each browser sends just a high-resolution stream, the server transcodes into low-resolution streams as required.

2.4.3.2. Additional Requirements

REQ-ID	DESCRIPTION
F16	The browser must support insertion of reference frames in outgoing media streams when requested by a peer.
F25	The browser must be able to render several concurrent audio and video streams.

3. Requirements Summary

3.1. General

This section contains the requirements on the browser derived from the use cases in Section 2.

Note: It is assumed that the user applications are executed on a browser. Whether the capabilities to implement specific browser requirements are implemented by the browser application, or are provided to the browser application by the underlying operating system, is outside the scope of this document.

3.2. Browser Requirements

Common, basic requirements	
REQ-ID	**DESCRIPTION**
F1	The browser must be able to use microphones and cameras as input devices to generate streams.
F2	The browser must be able to send streams and data to a peer in the presence of NATs.

F3	Transmitted streams and data must be rate controlled (meaning that the browser must, regardless of application behavior, reduce send rate when there is congestion).
F4	The browser must be able to receive, process, and render streams and data ("render" does not apply for data) from peers.
F5	The browser should be able to render good quality audio and video even in the presence of reasonable levels of jitter and packet losses.
F6	The browser must detect when a stream from a peer is not received anymore.
F7	When there are both incoming and outgoing audio streams, echo cancellation must be made available to avoid disturbing echo during conversation.
F8	The browser must support synchronization of audio and video.
F9	The browser should use encoding of streams suitable for the current rendering (e.g., video display size) and should change parameters if the rendering changes during the session
F10	The browser must support a baseline audio and video codec.
F11	It must be possible to protect streams and data from wiretapping [RFC2804] [RFC7258].
F12	The browser must enable verification, given the right circumstances and by use of other trusted communication, that streams and data received have not been manipulated by any party.

F13	The browser must encrypt, authenticate, and integrity protect media and data on a per-IP-packet basis, and it must drop incoming media and data packets that fail the per-IP-packet integrity check. In addition, the browser must support a mechanism for cryptographically binding media and data security keys to the user identity (see R-ID-BINDING in [RFC5479]).
F14	The browser must make it possible to set up a call between two parties without one party learning the other party's host IP address.
F15	The browser must be able to collect statistics, related to the transport of audio and video between peers, needed to estimate quality of experience.

Requirements related to network and topology

REQ-ID	DESCRIPTION
F16	The browser must support insertion of reference frames in outgoing media streams when requested by a peer.
F17	The communication session must survive across a change of the network interface used by the session.
F18	The browser must be able to send streams and data to a peer in the presence of NATs and firewalls that block UDP traffic.
F19	The browser must be able to use several STUN and TURN servers.
F20	The browser must support the use of STUN and TURN servers that are supplied by entities other than the web application (i.e., the network provider).
F21	The browser must be able to send streams and data to a peer in the presence of firewalls that only allow traffic via an HTTP Proxy, when firewall policy allows WebRTC traffic.

F22	The browser should be able to take advantage of available capabilities (supplied by network nodes) to differentiate voice, video, and data appropriately.

Requirements related to multiple peers and streams

REQ-ID	DESCRIPTION
F23	The browser must be able to transmit streams and data to several peers concurrently.
F24	The browser must be able to receive streams and data from multiple peers concurrently.
F25	The browser must be able to render several concurrent audio and video streams.
F26	The browser must be able to mix several audio streams.

Requirements related to audio processing

REQ-ID	DESCRIPTION
F27	The browser must be able to apply spatialization effects when playing audio streams.
F28	The browser must be able to measure the voice activity level in audio streams.
F29	The browser must be able to change the voice activity level in audio streams.
F30	The browser must be able to process and mix sound objects (media that is retrieved from another source than the established media stream(s) with the peer(s) with audio streams).

Requirements related to legacy interop

REQ-ID	DESCRIPTION
F31	The browser must support an audio media format (codec) that is commonly supported by existing telephony services.

F32	There should be a way to navigate a dual-tone multi-frequency signaling (DTMF) based Interactive Voice Response (IVR) system.
F33	The browser must be able to initiate and accept a media session where the data needed for establishment can be carried in SIP.

Other requirements

REQ-ID	DESCRIPTION
F34	The browser must be able to send short latency unreliable datagram traffic to a peer browser [RFC5405].
F35	The browser must be able to send reliable data traffic to a peer browser.
F36	The browser must be able to generate streams using the entire user display, a specific area of the user display or the information being displayed by a specific application.

4. Security Considerations

4.1. Introduction

A malicious web application might use the browser to perform Denial-of-Service (DoS) attacks on NAT infrastructure, or on peer devices. For example, a malicious web application might leak TURN credentials to unauthorized parties, allowing them to consume the TURN server's bandwidth. To address this risk, web applications should be prepared to revoke TURN credentials and issue new ones. Also, a malicious web application might silently establish outgoing, and accept incoming, streams on an already established connection.

Based on the identified security risks, this section will describe security considerations for the browser and web application.

4.2. Browser Considerations

The browser is expected to provide mechanisms for getting user consent to use device resources such as camera and microphone.

The browser is expected to provide mechanisms for informing the user that device resources such as camera and microphone are in use ("hot").

The browser must provide mechanisms for users to revise and even completely revoke consent to use device resources such as camera and microphone.

The browser is expected to provide mechanisms for getting user consent to use the screen (or a certain part of it) or what a certain application displays on the screen as source for streams.

The browser is expected to provide mechanisms for informing the user that the screen, part thereof, or an application is serving as a stream source ("hot").

The browser must provide mechanisms for users to revise and even completely revoke consent to use the screen, part thereof, or an application as a stream source.

The browser is expected to provide mechanisms in order to assure that streams are the ones the recipient intended to receive.

The browser is expected to provide mechanisms that allow the users to verify that the streams received have not be manipulated (F12).

The browser needs to ensure that media is not sent, and that received media is not rendered, until the associated stream establishment and handshake procedures with the remote peer have been successfully finished.

The browser needs to ensure that the stream negotiation procedures are not seen as DoS by other entities.

4.3. Web Application Considerations

The web application is expected to ensure user consent in sending and receiving media streams.

5. Normative References

[RFC2804] IAB and , "IETF Policy on Wiretapping", RFC 2804, May 2000, <http://www.rfc-editor.org/info/rfc2804>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010, <http://www.rfc-editor.org/info/rfc5245>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

[RFC5479] Wing, D., Ed., Fries, S., Tschofenig, H., and F. Audet, "Requirements and Analysis of Media Security Management Protocols", RFC 5479, April 2009, <http://www.rfc-editor.org/info/rfc5479>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, May 2014, <http://www.rfc-editor.org/info/rfc7258>.

Appendix A. API Requirements

This section contains the requirements on the API derived from the use cases in Section 2.

Note: As the W3C is responsible for the API, the API requirements in this specification are not normative.

REQ-ID	DESCRIPTION
A1	The web API must provide means for the application to ask the browser for permission to use cameras and microphones as input devices and to have access to the local file system.
A2	The web API must provide means for the web application to control how streams generated by input devices are used.
A3	The web API must provide means for the web application to control the local rendering of streams (locally generated streams and streams received from a peer).
A4	The web API must provide means for the web application to initiate the sending of a stream / stream components to a peer.
A5	The web API must provide means for the web application to control the media format (codec) to be used for the streams sent to a peer. Note: The level of control depends on whether the codec negotiation is handled by the browser or the web application.
A6	The web API must provide means for the web application to modify the media format for streams sent to a peer after a media stream has been established.
A7	The web API must provide means for informing the web application of whether or not the establishment of a stream with a peer was successful.

A8	The web API must provide means for the web application to mute/unmute a stream or stream component(s). When a stream is sent to a peer, mute status must be preserved in the stream received by the peer.
A9	The web API must provide means for the web application to cease the sending of a stream to a peer.
A10	The web API must provide means for the web application to cease the processing and rendering of a stream received from a peer.
A11	The web API must provide means for informing the web application when a stream from a peer is no longer received.
A12	The web API must provide means for informing the web application when high loss rates occur.
A13	The web API must provide means for the web application to apply spatialization effects to audio streams.
A14	The web API must provide means for the web application to detect the level in audio streams.
A15	The web API must provide means for the web application to adjust the level in audio streams.
A16	The web API must provide means for the web application to mix audio streams.
A17	The web API must provide a way to identify streams such that an application is able to match streams on a sending peer with the same stream on all receiving peers.
A18	The web API must provide a mechanism for sending and receiving isolated discrete chunks of data.

A19	The web API must provide means for the web application to indicate the type of audio signal (speech, audio) for audio stream(s) / stream component(s).
A20	It must be possible for an initiator or a responder web application to indicate the types of media it is willing to accept incoming streams for when setting up a connection (audio, video, other). The types of media to be accepted can be a subset of the types of media the browser is able to accept.
A21	The web API must provide means for the application to ask the browser for permission to use the screen, a certain area on the screen, or what a certain application displays on the screen as input to streams.
A22	The web API must provide means for the application to specify several STUN and/or TURN servers to use.
A23	The web API must provide means for the application to specify the priority to apply for outgoing streams and data.
A24	The web API must provide a mechanism for sending and receiving files.
A25	It must be possible for the application to instruct the browser to refrain from exposing the host IP address to the application.
A26	The web API must provide means for the application to obtain the statistics (related to transport, and collected by the browser) needed to estimate the quality of service.

Acknowledgements

The authors wish to thank Bernard Aboba, Gunnar Hellstrom, Martin Thomson, Lars Eggert, Matthew Kaufman, Emil Ivov, Eric Rescorla, Eric Burger, John Leslie, Dan Wing, Richard Barnes, Barry Dingle, Dale Worley, Ted Hardie, Mary Barnes, Dan Burnett, Stephan Wenger, Harald Alvestrand, Cullen Jennings, Andrew Hutton and everyone else in the RTCWEB community that have provided comments, feedback, text and improvement proposals on the document. A big thank you to everyone that provided comments as part of the IESG evaluation and to everyone else that provided comments and input in order to improve the document.

Authors' Addresses

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: christer.holmberg@ericsson.com

Stefan Hakansson
Ericsson
Laboratoriegrand 11
Lulea 97128
Sweden

EMail: stefan.lk.hakansson@ericsson.com

Goran AP Eriksson
Ericsson
Farogatan 6
Stockholm 16480
Sweden

EMail: goran.ap.eriksson@ericsson.com