wish
Internet-Draft
Intended status: Standards Track
Expires: 19 February 2026

S. Murillo
Millicast
C. Chen
ByteDance
D. Jenkins, Ed.
Everycast Labs Ltd
18 August 2025

WebRTC-HTTP Egress Protocol (WHEP)
draft-ietf-wish-whep-03

Abstract

This document describes a simple HTTP-based protocol that will allow WebRTC-based viewers to watch content from streaming services and/or Content Delivery Networks (CDNs) or WebRTC Transmission Network (WTNs).

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 19 February 2026.

Copyright Notice

Copyright (c) 2025 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 (https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The IETF RTCWEB working group standardized JSEP ([RFC9429]), a mechanism used to control the setup, management, and teardown of a multimedia session. It also describes how to negotiate media flows using the Offer/Answer Model with the Session Description Protocol (SDP) [RFC3264] including the formats for data sent over the wire (e.g., media types, codec parameters, and encryption). WebRTC intentionally does not specify a signaling transport protocol at application level.

While WebRTC can be integrated with standard signaling protocols like SIP [RFC3261] or XMPP [RFC6120], they are not designed to be used in broadcasting/streaming services, and there also is no sign of adoption in that industry. RTSP [RFC7826], which is based on RTP, does not support the SDP offer/answer model [RFC3264] for negotiating the characteristics of the media session.

There are many situations in which the lack of a standard protocol for consuming media from streaming service using WebRTC has become a problem:

* Interoperability between WebRTC services and products.

* Reusing player software which can be integrated easily.

* Integration with Dynamic Adaptive Streaming over HTTP (DASH) for offering live streams via WebRTC while offering a time-shifted version via DASH.

* Playing WebRTC streams on devices that don't support custom javascript to be run (like TVs).

This document mimics what has been done in the WebRTC HTTP Ingest Protocol (WHIP) [I-D.draft-ietf-wish-whip] for ingestion and specifies a simple HTTP-based protocol that can be used for consuming media from a streaming service using WebRTC.

2. 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.

3. Overview

The WebRTC-HTTP Egress Protocol (WHEP) is designed to facilitate an exchange of Session Description Protocol (SDP) offers and answers using HTTP POST requests. This exchange is a fundamental step in establishing an Interactive Connectivity Establishment (ICE) and Datagram Transport Layer Security (DTLS) session between WHEP player and the streaming service endpoint (Media Server).

Upon successful establishment of the ICE/DTLS session, unidirectional media data transmission commences from the media server to the WHEP player. It is important to note that SDP renegotiations are not supported in WHEP, meaning that no modifications to the "m=" sections can be made after the initial SDP offer/answer exchange via HTTP POST is completed and only ICE related information can be updated via HTTP PATCH requests as defined in Section 4.4.

The WHEP player always initiates the streaming session by sending an SDP offer to the WHEP endpoint. The WHEP endpoint can then choose to either accept the client’s offer by responding with an SDP answer, or reject the client’s offer and counter with its own SDP offer. If the WHEP endpoint sends a counter-offer, the client must then respond with an SDP answer. A WHEP player must support processing both SDP answers (when the WHEP endpoint accepts the client’s offer) and SDP offers (when the WHEP endpoint sends a counter-offer) in response to the initial request.

3.1. Protocol Operation Flow

The following diagram illustrates the core operation of the WHEP protocol for initiating and terminating a viewing session:

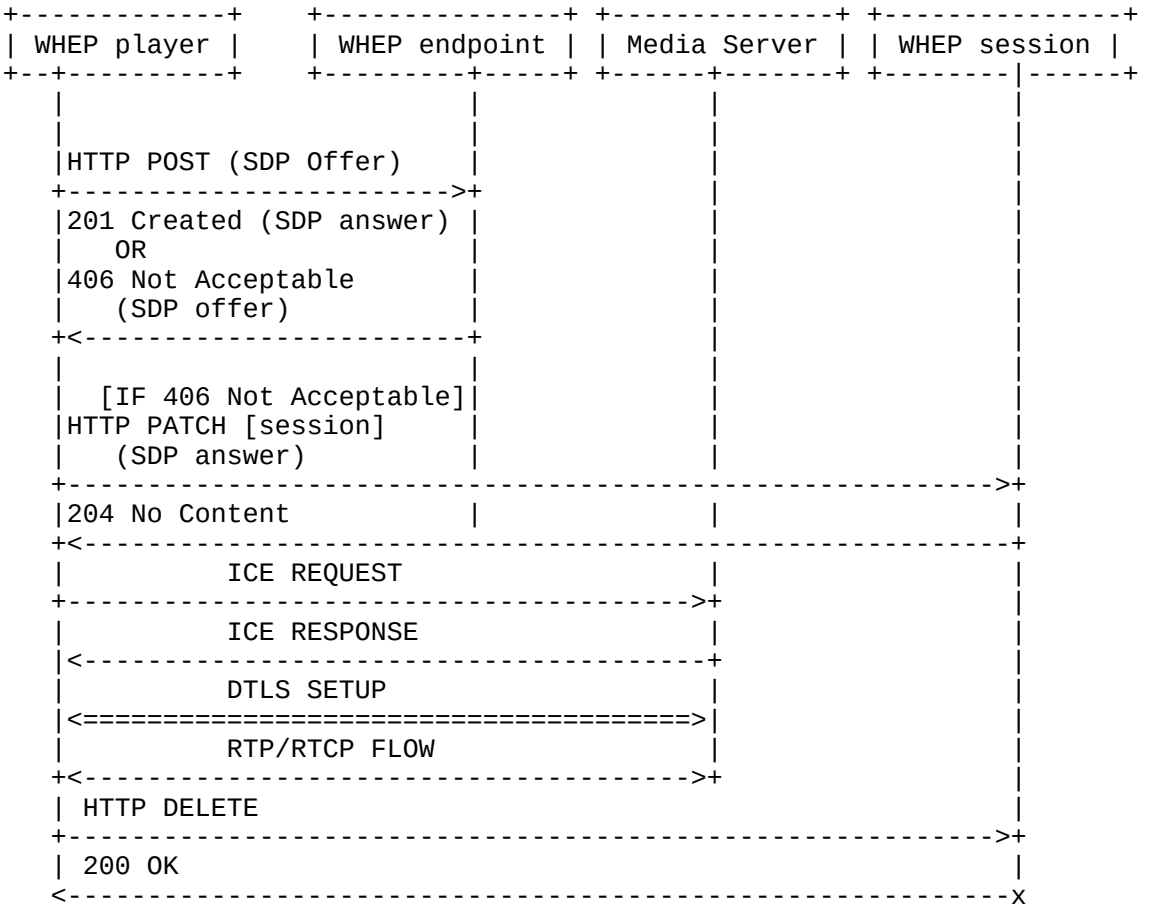

Figure 1: WHEP session setup and teardown

The elements in Figure 1 are described as follows:

* WHEP player: This represents the WebRTC media player, which functions as a client of the WHEP protocol by receiving and decoding the media from a remote media server.

* WHEP endpoint: This denotes the egress server that receives the initial WHEP request.

* WHEP endpoint URL: Refers to the URL of the WHEP endpoint responsible for creating the WHEP session.

* Media server: This is the WebRTC Media Server that establishes the media session with the WHEP player and delivers the media to it.

* WHEP session: Indicates the allocated HTTP resource by the WHEP endpoint for an ongoing egress session.

* WHEP session URL: Refers to the URL of the WHEP resource allocated by the WHEP endpoint for a specific media session. The WHEP player can send requests to the WHEP session using this URL to modify the session, such as ICE operations or termination.

The Figure 1 illustrates the communication flow between a WHEP player, WHEP endpoint, media server, and WHEP session. This flow outlines the process of setting up and tearing down a playback session using the WHEP protocol, involving negotiation, ICE for Network Address Translation (NAT) traversal, DTLS and Secure Real-time Transport Protocol (SRTP) for security, and RTP/RTCP for media transport:

3.2. Protocol Operation Steps

* WHEP player: Initiates the communication by sending an HTTP POST with an SDP offer to the WHEP endpoint.

* WHEP endpoint: Responds with either a "201 Created" message containing an SDP answer (accepting the client’s offer) or a "406 Not Acceptable" message containing an SDP counter-offer (rejecting the client’s offer).

* WHEP player: If the WHEP endpoint responded with "406 Not Acceptable", the player sends an HTTP PATCH containing an SDP answer to the WHEP session URL.

* WHEP session: If applicable, responds with a "204 No Content" message to the PATCH request.

* WHEP player and media server: Establish ICE and DTLS sessions for NAT traversal and secure communication.

* RTP/RTCP Flow: Real-time Transport Protocol and Real-time Transport Control Protocol flows are established for media transmission from the media server to the WHEP player, secured by the SRTP profile.

* WHEP player: Sends an HTTP DELETE to terminate the WHEP session.

* WHEP session: Responds with a "200 OK" to confirm the session termination.

4. Protocol Operation

4.1. HTTP usage

Following [BCP56] guidelines, WHEP players MUST NOT match error codes returned by the WHEP endpoints and resources to a specific error cause indicated in this specification. WHEP players MUST be able to handle all applicable status codes gracefully falling back to the generic n00 semantics of a given status code on unknown error codes. WHEP endpoints and resources could convey finer-grained error information by a problem statement json object in the response message body of the failed request as per [RFC9457].

The WHEP endpoints and sessions are origin servers as defined in Section 3.6. of [RFC9110] handling the requests and providing responses for the underlying HTTP resources. Those HTTP resources do not have any representation defined in this specification, so the WHEP endpoints and sessions MUST return a 2XX successful response with no content when a GET request is received.

4.2. Playback session set up

In order to set up a streaming session, the WHEP player MUST generate an SDP offer according to the JSEP rules for an initial offer as in Section 5.2.1 of [RFC9429] and perform an HTTP POST request as per Section 9.3.3 of [RFC9110] to the configured WHEP endpoint URL.

The HTTP POST request MUST have a content type of "application/sdp" and contain the SDP offer as the body. Upon receiving the HTTP POST request, the WHEP endpoint can choose to either accept the client’s offer or reject it in favor of sending its own offer.

4.2.1. Server Accepts Client Offer

If the WHEP endpoint chooses to accept the client’s SDP offer, it MUST generate an SDP answer according to the JSEP rules for an initial answer as in Section 5.3.1 of [RFC9429] and return a "201 Created" response with a content type of "application/sdp", the SDP answer as the body, and a Location header field pointing to the newly created WHEP session.

4.2.2. Server Sends Counter-offer

If the WHEP endpoint chooses to reject the client’s SDP offer, it MUST generate its own SDP offer according to the JSEP rules for an initial offer as in Section 5.2.1 of [RFC9429] and return a "406 Not Acceptable" response with a content type of "application/sdp", the SDP counter-offer as the body, and a Location header field pointing to the WHEP session resource that will be created upon completion of the offer/answer exchange.

The WHEP endpoint MAY include a "valid-until" parameter in the Content-Type header to indicate how long the counter-offer remains valid. If no "valid-until" parameter is provided, the counter-offer remains valid for 30 seconds from the time the response was sent. The "valid-until" parameter value MUST be an HTTP-date as defined in Section 5.6.7 of [RFC9110].

When the WHEP player receives a counter-offer from the WHEP endpoint, it MUST generate an SDP answer according to the JSEP rules for an initial answer as in Section 5.3.1 of [RFC9429]. To send the SDP answer, the WHEP player MUST perform an HTTP PATCH request as per [RFC5789] to the WHEP session URL with content type of "application/sdp" and the SDP answer as the body. The WHEP endpoint MUST return a "204 No Content" response. If the SDP is malformed, the WHEP endpoint MUST reject the HTTP PATCH request with an appropriate 4XX error response.

4.2.3. Determining Server Response Type

WHEP players can determine the WHEP endpoint’s response type by examining the HTTP status code:

* *"201 Created"*: The WHEP endpoint has accepted the client’s offer and responded with an SDP answer. The WHEP session has been created and is ready for media transmission.

* *"406 Not Acceptable"*: The WHEP endpoint has rejected the client’s offer and responded with an SDP counter-offer. The client MUST send an HTTP PATCH request to the WHEP session URL with an SDP answer to complete the session establishment.

4.2.4. Error Conditions

If the HTTP POST to the WHEP endpoint has a content type different than "application/sdp" or the SDP is malformed, the WHEP endpoint MUST reject the HTTP POST request with an appropriate 4XX error response.

4.2.5. Media Direction Attributes

As the WHEP protocol only supports the playback use case with unidirectional media:

* When a WHEP player sends an SDP offer, it SHOULD use "recvonly" attribute but MAY use the "sendrecv" attribute instead. The "inactive" and "sendonly" attributes MUST NOT be used.

* When a WHEP endpoint sends an SDP answer (accepting client offer), it MUST use "sendonly" attribute in the SDP answer.

* When a WHEP endpoint sends an SDP counter-offer, it SHOULD use "sendonly" attribute but MAY use the "sendrecv" attribute instead. The "inactive" and "recvonly" attributes MUST NOT be used.

* When a WHEP player sends an SDP answer (responding to server counter-offer), it MUST use "recvonly" attribute in the SDP answer.

4.2.6. Codec Recommendations

WHEP players SHOULD include as many supported codecs as possible in their SDP offers and answers to maximize compatibility and enable dynamic streaming scenarios. This applies whether the WHEP player is sending the initial offer or responding to a server counter-offer with an answer.

Including a comprehensive list of supported codecs enables several important use cases:

* *Dynamic source switching*: A WHEP endpoint may need to change which camera or media source a stream is connected to, potentially requiring different codecs for optimal quality or performance.

* *Adaptive streaming*: The WHEP endpoint may switch between different codec configurations based on network conditions, viewer capabilities, or content characteristics.

* *Failover scenarios*: If the primary codec encounters issues, having alternative codecs available allows seamless fallback without requiring renegotiation.

* *Multi-resolution support*: Different codecs may be optimal for different resolutions or bitrates that the WHEP endpoint may need to provide.

WHEP players that restrict their codec offerings may prevent these advanced streaming scenarios and limit the WHEP endpoint’s ability to provide optimal streaming experiences.

4.2.7. Examples

Following Figure 2 is an example where the WHEP endpoint accepts the client’s offer:

```
POST /whep/endpoint HTTP/1.1
Host: whep.example.com
Content-Type: application/sdp
Content-Length: 1326

v=0
o=- 5228595038118931041 2 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0 1
a=extmap-allow-mixed
a=ice-options:trickle ice2
m=audio 9 UDP/TLS/RTP/SAVPF 111
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:zjkk
a=ice-pwd:bP+XJMM09aR8AiX1jdukzR6Y
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58:29:ED:77:2A:0D:2
4:AE:ED:AD:30:BC:BD:F1:9C:02
a=setup:actpass
a=mid:0
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=recvonly
a=rtcp-mux
a=rtcp-mux-only
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
m=video 0 UDP/TLS/RTP/SAVPF 96 97
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:zjkk
a=ice-pwd:bP+XJMM09aR8AiX1jdukzR6Y
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58:29:ED:77:2A:0D:2
4:AE:ED:AD:30:BC:BD:F1:9C:02
a=setup:actpass
a=mid:1
a=bundle-only
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:10 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=extmap:11 urn:ietf:params:rtp-hdrext:sdes:repaired-rtp-stream-id
a=recvonly
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=rtpmap:96 VP8/90000
a=rtcp-fb:96 ccm fir
a=rtcp-fb:96 nack
a=rtcp-fb:96 nack pli
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96
```

```
HTTP/1.1 201 Created
ETag: "xyzzy"
Content-Type: application/sdp
Content-Length: 1400
Location: https://whep.example.org/sessions/id

v=0
o=- 1657793490019 1 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0 1
a=extmap-allow-mixed
a=ice-lite
a=ice-options:trickle ice2
m=audio 9 UDP/TLS/RTP/SAVPF 111
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:526be20a538ee422
a=ice-pwd:2e13dde17c1cb009202f627fab90cbec358d766d049c9697
a=fingerprint:sha-256 F7:EB:F3:3E:AC:D2:EA:A7:C1:EC:79:D9:B3:8A:35:DA:70:86:4F:46:D9:2D:C
C:D0:BC:81:9F:67:EF:34:2E:BD
a=candidate:1 1 UDP 2130706431 198.51.100.1 39132 typ host
a=setup:passive
a=mid:0
a=bundle-only
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=sendonly
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
a=msid:- d46fb922-d52a-4e9c-aa87-444eadc1521b
m=video 0 UDP/TLS/RTP/SAVPF 96 97
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:526be20a538ee422
a=ice-pwd:2e13dde17c1cb009202f627fab90cbec358d766d049c9697
a=fingerprint:sha-256 F7:EB:F3:3E:AC:D2:EA:A7:C1:EC:79:D9:B3:8A:35:DA:70:86:4F:46:D9:2D:C
C:D0:BC:81:9F:67:EF:34:2E:BD
a=candidate:1 1 UDP 2130706431 198.51.100.1 39132 typ host
a=setup:passive
a=mid:1
a=bundle-only
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:10 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=extmap:11 urn:ietf:params:rtp-hdrext:sdes:repaired-rtp-stream-id
a=sendonly
a=rtcp-mux
a=rtcp-mux-only
```

```
a=rtcp-rsize
a=rtpmap:96 VP8/90000
a=rtcp-fb:96 ccm fir
a=rtcp-fb:96 nack
a=rtcp-fb:96 nack pli
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96
a=msid:- d46fb922-d52a-4e9c-aa87-444eadc1521b
```

Figure 2: Example where WHEP endpoint accepts client offer

Following Figure 3 is an example where the WHEP endpoint sends a counter-offer:

```
POST /channel/teeny-tasty-crayon HTTP/1.1
Host: whep.example.com
Content-Type: application/sdp
Content-Length: 1326

v=0
o=- 5228595038118931041 2 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0 1
a=extmap-allow-mixed
a=ice-options:trickle ice2
m=audio 9 UDP/TLS/RTP/SAVPF 111
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:zjkk
a=ice-pwd:bP+XJMM09aR8AiX1jdukzR6Y
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58:29:ED:77:2A:0D:2
4:AE:ED:AD:30:BC:BD:F1:9C:02
a=setup:actpass
a=mid:0
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=recvonly
a=rtcp-mux
a=rtcp-mux-only
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
m=video 0 UDP/TLS/RTP/SAVPF 96 97
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:zjkk
a=ice-pwd:bP+XJMM09aR8AiX1jdukzR6Y
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58:29:ED:77:2A:0D:2
4:AE:ED:AD:30:BC:BD:F1:9C:02
a=setup:actpass
a=mid:1
```

```
a=bundle-only
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:10 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=extmap:11 urn:ietf:params:rtp-hdrext:sdes:repaired-rtp-stream-id
a=recvonly
a=rtcp-mux
a=rtcp-mux-only
a=rtcp-rsize
a=rtpmap:96 VP8/90000
a=rtcp-fb:96 ccm fir
a=rtcp-fb:96 nack
a=rtcp-fb:96 nack pli
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96

HTTP/1.1 406 Not Acceptable
Content-Type: application/sdp; valid-until="Wed, 09 Oct 2024 10:00:00 GMT"
Content-Length: 3552
Location: https://whep.example.com/channel/teeny-tasty-crayon/3de3c94a-fc0f-4659-bcaf-8bd
ebf718457

v=0
o=- 2438602337097565327 2 IN IP4 127.0.0.1
s=-
t=0 0
a=msid-semantic: WMS feedbackvideomslabel e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6
a=group:BUNDLE 0 1 2 3
m=video 9 RTP/SAVPF 100 96
c=IN IP4 0.0.0.0
a=rtpmap:100 VP8/90000
a=rtpmap:96 rtx/90000
a=fmtp:96 apt=100
a=rtcp:9 IN IP4 0.0.0.0
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=setup:active
a=mid:0
a=sendonly
a=ice-ufrag:CiYfXaM3jHrmpF
a=ice-pwd:VQFGPhTQj/BnaJ/tkec9m1Hi
a=fingerprint:sha-256 4C:C3:25:E0:29:75:AF:01:53:94:CD:C4:6F:5F:15:5E:E3:1A:10:AE:8C:96:0
7:5A:18:AC:49:5F:55:68:6C:C5
a=candidate:676201573392 1 udp 142541055 172.234.108.130 10000 typ host generation 0 netw
ork-id 1
a=ssrc:3592962548 cname:feedbackvideocname
a=ssrc:3592962548 label:feedbackvideolabel
a=ssrc:3592962548 mslabel:feedbackvideomslabel
a=ssrc:3592962548 msid:feedbackvideomslabel feedbackvideolabel
a=rtcp-mux
m=application 9 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
```

```
a=rtcp:9 IN IP4 0.0.0.0
a=setup:active
a=mid:1
a=sendonly
a=ice-ufrag:CiYfXaM3jHrmpF
a=ice-pwd:VQFGPhTQj/BnaJ/tkec9m1Hi
a=fingerprint:sha-256 4C:C3:25:E0:29:75:AF:01:53:94:CD:C4:6F:5F:15:5E:E3:1A:10:AE:8C:96:0
7:5A:18:AC:49:5F:55:68:6C:C5
a=candidate:676201573392 1 udp 142541055 172.234.108.130 10000 typ host generation 0 netw
ork-id 1
a=rtcp-mux
a=sctpmap:5000 webrtc-datachannel 262144
m=audio 9 RTP/SAVPF 111
c=IN IP4 0.0.0.0
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
a=rtcp:9 IN IP4 0.0.0.0
a=extmap:1 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:8 c9:params:rtp-hdrext:info
a=setup:active
a=mid:2
a=sendonly
a=ice-ufrag:CiYfXaM3jHrmpF
a=ice-pwd:VQFGPhTQj/BnaJ/tkec9m1Hi
a=fingerprint:sha-256 4C:C3:25:E0:29:75:AF:01:53:94:CD:C4:6F:5F:15:5E:E3:1A:10:AE:8C:96:0
7:5A:18:AC:49:5F:55:68:6C:C5
a=candidate:676201573392 1 udp 142541055 172.234.108.130 10000 typ host generation 0 netw
ork-id 1
a=ssrc:2338673210 cname:0p6mZhWJw+/818iW
a=ssrc:2338673210 label:2fcad988-9bc2-4705-b408-9aee41bc3d71
a=ssrc:2338673210 mslabel:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6
a=ssrc:2338673210 msid:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6 2fcad988-9bc2-4705-b408-9aee4
1bc3d71
a=rtcp-mux
m=video 9 RTP/SAVPF 100 96
c=IN IP4 0.0.0.0
a=rtpmap:100 VP8/90000
a=rtpmap:96 rtx/90000
a=fmtp:96 apt=100
a=rtcp:9 IN IP4 0.0.0.0
a=rtcp-fb:100 goog-remb
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=setup:active
a=mid:3
a=sendonly
a=ice-ufrag:CiYfXaM3jHrmpF
a=ice-pwd:VQFGPhTQj/BnaJ/tkec9m1Hi
a=fingerprint:sha-256 4C:C3:25:E0:29:75:AF:01:53:94:CD:C4:6F:5F:15:5E:E3:1A:10:AE:8C:96:0
7:5A:18:AC:49:5F:55:68:6C:C5
a=candidate:676201573392 1 udp 142541055 172.234.108.130 10000 typ host generation 0 netw
ork-id 1
```

```
a=ssrc:755359452 cname:0p6mZhWJw+/818iW
a=ssrc:755359452 label:d6bca5d1-b69d-4d9d-8b5d-9117707cdb81
a=ssrc:755359452 mslabel:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6
a=ssrc:755359452 msid:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6 d6bca5d1-b69d-4d9d-8b5d-911770
7cdb81
a=ssrc:280880788 cname:0p6mZhWJw+/818iW
a=ssrc:280880788 label:d6bca5d1-b69d-4d9d-8b5d-9117707cdb81
a=ssrc:280880788 mslabel:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6
a=ssrc:280880788 msid:e6ddf4a9-b5ed-4e87-9ae3-ef126a9164d6 d6bca5d1-b69d-4d9d-8b5d-911770
7cdb81
a=ssrc-group:FID 755359452 280880788
a=rtcp-mux

PATCH /channel/teeny-tasty-crayon/3de3c94a-fc0f-4659-bcaf-8bdebf718457 HTTP/1.1
Host: whep.example.com
Content-Type: application/sdp
Content-Length: 2410

v=0
o=- 4541478638207698795 2 IN IP4 127.0.0.1
s=-
t=0 0
a=group:BUNDLE 0 1 2 3
a=msid-semantic: WMS
m=video 56464 RTP/SAVPF 100 96
c=IN IP4 192.168.167.137
a=rtcp:9 IN IP4 0.0.0.0
a=candidate:170904481 1 udp 2122129151 192.168.167.137 56464 typ host generation 0 networ
k-id 1 network-cost 10
a=candidate:3499970512 1 udp 2122265343 fd2e:9c8b:abe4:2:838:1bbe:9d48:3ec 53930 typ host
 generation 0 network-id 3 network-cost 10
a=candidate:3061500384 1 udp 2122197247 2001:9b1:28fe:9400:88fb:57a4:5888:153b 62309 typ
host generation 0 network-id 2 network-cost 10
a=ice-ufrag:37nK
a=ice-pwd:NZH/oQX6FHAl+EmWvpgoPZzC
a=ice-options:trickle
a=fingerprint:sha-256 00:91:87:75:0D:C7:F6:D4:65:4D:9F:1D:EF:52:A1:60:02:8D:E7:67:73:68:B
9:78:12:D9:FD:3E:09:F8:BF:3D
a=setup:passive
a=mid:0
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=recvonly
a=rtcp-mux
a=rtpmap:100 VP8/90000
a=rtpmap:96 rtx/90000
a=fmtp:96 apt=100
m=application 9 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 0.0.0.0
a=ice-ufrag:37nK
a=ice-pwd:NZH/oQX6FHAl+EmWvpgoPZzC
a=ice-options:trickle
a=fingerprint:sha-256 00:91:87:75:0D:C7:F6:D4:65:4D:9F:1D:EF:52:A1:60:02:8D:E7:67:73:68:B
9:78:12:D9:FD:3E:09:F8:BF:3D
a=setup:passive
```

```
a=mid:1
a=sctp-port:5000
m=audio 9 RTP/SAVPF 111
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:37nK
a=ice-pwd:NZH/oQX6FHAl+EmWvpgoPZzC
a=ice-options:trickle
a=fingerprint:sha-256 00:91:87:75:0D:C7:F6:D4:65:4D:9F:1D:EF:52:A1:60:02:8D:E7:67:73:68:B
9:78:12:D9:FD:3E:09:F8:BF:3D
a=setup:passive
a=mid:2
a=extmap:1 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=recvonly
a=rtcp-mux
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
m=video 9 RTP/SAVPF 100 96
c=IN IP4 0.0.0.0
a=rtcp:9 IN IP4 0.0.0.0
a=ice-ufrag:37nK
a=ice-pwd:NZH/oQX6FHAl+EmWvpgoPZzC
a=ice-options:trickle
a=fingerprint:sha-256 00:91:87:75:0D:C7:F6:D4:65:4D:9F:1D:EF:52:A1:60:02:8D:E7:67:73:68:B
9:78:12:D9:FD:3E:09:F8:BF:3D
a=setup:passive
a=mid:3
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=recvonly
a=rtcp-mux
a=rtpmap:100 VP8/90000
a=rtcp-fb:100 goog-remb
a=rtcp-fb:100 nack
a=rtcp-fb:100 nack pli
a=rtpmap:96 rtx/90000
a=fmtp:96 apt=100

HTTP/1.1 204 No Content
```

Figure 3: Example where WHEP endpoint sends counter-offer

4.2.8. Session Management

The WHEP endpoint COULD require a live publishing to be happening in order to allow a WHEP players to start viewing a stream. In that case, the WHEP endpoint SHALL return a "409 Conflict" response to the POST request issued by the WHEP player with a "Retry-After" header indicating the number of seconds before sending a new request. WHEP players MAY periodically try to connect to the WHEP session with exponential backoff period with an initial value of the "Retry-After" header value in the "409 Conflict" response.

Once a session is setup, consent freshness as per [RFC7675] SHALL be used to detect non-graceful disconnection by full ICE implementations and DTLS teardown for session termination by either side.

4.3. Playback session termination

To explicitly terminate a WHEP session, the WHEP player MUST perform an HTTP DELETE request to the WHEP session URL returned in the Location header field of the initial HTTP POST. Upon receiving the HTTP DELETE request, the WHEP session will be removed and the resources freed on the media server, terminating the ICE and DTLS sessions.

A media server terminating a session MUST follow the procedures in Section 5.2 of [RFC7675] for immediate revocation of consent.

The WHEP endpoints MUST support OPTIONS requests for Cross-Origin Resource Sharing (CORS) as defined in [FETCH]. The "200 OK" response to any OPTIONS request SHOULD include an "Accept-Post" header with a media type value of "application/sdp" as per [W3C.REC-ldp-20150226].

4.4. ICE support

ICE [RFC8445] is a protocol addressing the complexities of NAT traversal, commonly encountered in Internet communication. NATs hinder direct communication between devices on different local networks, posing challenges for real-time applications. ICE facilitates seamless connectivity by employing techniques to discover and negotiate efficient communication paths.

Trickle ICE [RFC8838] optimizes the connectivity process by incrementally sharing potential communication paths, reducing latency, and facilitating quicker establishment.

ICE Restarts are crucial for maintaining connectivity in dynamic network conditions or disruptions, allowing devices to re-establish communication paths without complete renegotiation. This ensures minimal latency and reliable real-time communication.

Trickle ICE and ICE restart support are RECOMMENDED for both WHEP sessions and clients.

4.4.1. HTTP PATCH request usage

The WHEP player MAY perform trickle ICE or ICE restarts by sending an HTTP PATCH request as per [RFC5789] to the WHEP session URL, with a body containing a SDP fragment with media type "application/trickle-ice-sdpfrag" as specified in [RFC8840] carrying the relevant ICE information. If the HTTP PATCH to the WHEP session has a content type different than "application/trickle-ice-sdpfrag" or the SDP fragment is malformed, the WHEP session MUST reject the HTTP PATCH with an appropriate 4XX error response.

If the WHEP session supports either Trickle ICE or ICE restarts, but not both, it MUST return a "422 Unprocessable Content" error response for the HTTP PATCH requests that are not supported as per Section 15.5.21 of [RFC9110].

The WHEP player MAY send overlapping HTTP PATCH requests to one WHEP session. Consequently, as those HTTP PATCH requests may be received out-of-order by the WHEP session, if WHEP session supports ICE restarts, it MUST generate a unique strong entity-tag identifying the ICE session as per Section 8.8.3 of [RFC9110], being OPTIONAL otherwise. The initial value of the entity-tag identifying the initial ICE session MUST be returned in an ETag header field in the "201 Created" response to the initial POST request to the WHEP endpoint.

WHEP players SHOULD NOT use entity-tag validation when matching a specific ICE session is not required, such as for example when initiating a DELETE request to terminate a session. WHEP sessions MUST ignore any entity-tag value sent by the WHEP player when ICE session matching is not required, as in the HTTP DELETE request.

Missing or outdated ETags in the PATCH requests from WHEP players will be answered by WHEP sessions as per Section 13.1.1 of [RFC9110] and Section 3 of [RFC6585], with a "428 Precondition Required" response for a missing entity tag, and a "412 Precondition Failed" response for a non-matching entity tag.

4.4.2. Trickle ICE

Depending on the Trickle ICE support on the WHEP player, the initial offer by the WHEP player MAY be sent after the full ICE gathering is complete with the full list of ICE candidates, it MAY only contain local candidates as per [RFC8445] or even an empty list of candidates as per [RFC8863]. For the purpose of reducing setup times, when using Trickle ICE the WHEP player SHOULD send the SDP offer as soon as possible, containing either locally gathered ICE candidates or an empty list of candidates.

In order to simplify the protocol, the WHEP session cannot signal additional ICE candidates to the WHEP player after the SDP answer has been sent. The WHEP endpoint SHALL gather all the ICE candidates for the media server before responding to the client request and the SDP answer SHALL contain the full list of ICE candidates of the media server.

As the WHEP player needs to know the WHEP session URL associated with the ICE session in order to send a PATCH request containing new ICE candidates, it MUST wait and buffer any gathered candidates until the "201 Created" HTTP response to the initial POST request is received. In order to lower the HTTP traffic and processing time required the WHEP player SHOULD send a single aggregated HTTP PATCH request with all the buffered ICE candidates once the response is received. Additionally, if ICE restarts are supported by the WHEP session, the WHEP player needs to know the entity-tag associated with the ICE session in order to send a PATCH request containing new ICE candidates, so it MUST also wait and buffer any gathered candidates until it receives the HTTP response with the new entity-tag value to the last PATCH request performing an ICE restart.

WHEP players generating the HTTP PATCH body with the SDP fragment and its subsequent processing by WHEP sessions MUST follow to the guidelines defined in Section 4.4 of [RFC8840] with the following considerations:

* As per [RFC9429], only m-sections not marked as bundle-only can gather ICE candidates, so given that the "max-bundle" policy is being used, the SDP fragment will contain only the offerer-tagged m-line of the bundle group.

* The WHEP player MAY exclude ICE candidates from the HTTP PATCH body if they have already been confirmed by the WHEP session with a successful HTTP response to a previous HTTP PATCH request.

WHEP sessions and players that support Trickle ICE MUST make use of entity-tags and conditional requests as explained in Section 4.4.1.

When a WHEP session receives a PATCH request that adds new ICE candidates without performing an ICE restart, it MUST return a "204 No Content" response without a body and MUST NOT include an ETag header in the response. If the WHEP session does not support a candidate transport or is not able to resolve the connection address, it MUST silently discard the candidate and continue processing the rest of the request normally.

```
PATCH /session/id HTTP/1.1
Host: whep.example.com
If-Match: "xyzzy"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 576

a=group:BUNDLE 0 1
m=audio 9 UDP/TLS/RTP/SAVPF 111
a=mid:0
a=ice-ufrag:EsAw
a=ice-pwd:P2uYro0UCOQ4zxjKXaWCBui1
a=candidate:1387637174 1 udp 2122260223 192.0.2.1 61764 typ host generation 0 ufrag EsAw
network-id 1
a=candidate:3471623853 1 udp 2122194687 198.51.100.2 61765 typ host generation 0 ufrag Es
Aw network-id 2
a=candidate:473322822 1 tcp 1518280447 192.0.2.1 9 typ host tcptype active generation 0 u
frag EsAw network-id 1
a=candidate:2154773085 1 tcp 1518214911 198.51.100.2 9 typ host tcptype active generation
 0 ufrag EsAw network-id 2
a=end-of-candidates

HTTP/1.1 204 No Content
```

Figure 4: Example of a Trickle ICE request and response

Figure 4 shows an example of the Trickle ICE procedure where the WHEP player sends a PATCH request with updated ICE candidate information and receives a successful response from the WHEP session.

4.4.3. ICE Restarts

As defined in [RFC8839], when an ICE restart occurs, a new SDP offer/answer exchange is triggered. However, as WHEP does not support renegotiation of non-ICE related SDP information, a WHEP player will not send a new offer when an ICE restart occurs. Instead, the WHEP player and WHEP session will only exchange the relevant ICE information via an HTTP PATCH request as defined in Section 4.4.1 and MUST assume that the previously negotiated non-ICE related SDP information still apply after the ICE restart.

When performing an ICE restart, the WHEP player MUST include the updated "ice-pwd" and "ice-ufrag" in the SDP fragment of the HTTP PATCH request body as well as the new set of gathered ICE candidates as defined in [RFC8840]. Similar what is defined in Section 4.4.2,

as per [RFC9429] only m-sections not marked as bundle-only can gather ICE candidates, so given that the "max-bundle" policy is being used, the SDP fragment will contain only the offerer-tagged m-line of the bundle group. A WHEP player sending a PATCH request for performing ICE restart MUST contain an "If-Match" header field with a field-value "*" as per Section 13.1.1 of [RFC9110].

[RFC8840] states that an agent MUST discard any received requests containing "ice-pwd" and "ice-ufrag" attributes that do not match those of the current ICE Negotiation Session, however, any WHEP session receiving an updated "ice-pwd" and "ice-ufrag" attributes MUST consider the request as performing an ICE restart instead and, if supported, SHALL return a "200 OK" with an "application/trickle-ice-sdpfrag" body containing the new ICE username fragment and password and a new set of ICE candidates for the WHEP session. Also, the "200 OK" response for a successful ICE restart MUST contain the new entity-tag corresponding to the new ICE session in an ETag response header field and MAY contain a new set of ICE candidates for the media server.

As defined in Section 4.4.1.1.1 of [RFC8839] the set of candidates after an ICE restart may include some, none, or all of the previous candidates for that data stream and may include a totally new set of candidates. So after performing a successful ICE restart, both the WHEP player and the WHEP session MUST replace the previous set of remote candidates with the new set exchanged in the HTTP PATCH request and response, discarding any remote ICE candidate not present on the new set. Both the WHEP player and the WHEP session MUST ensure that the HTTP PATCH requests and response bodies include the same ’ice-options,’ ’ice-pacing,’ and ’ice-lite’ attributes as those used in the SDP offer or answer.

If the ICE restart request cannot be satisfied by the WHEP session, the resource MUST return an appropriate HTTP error code and MUST NOT terminate the session immediately and keep the existing ICE session. The WHEP player MAY retry performing a new ICE restart or terminate the session by issuing an HTTP DELETE request instead. In any case, the session MUST be terminated if the ICE consent expires as a consequence of the failed ICE restart as per Section 5.1 of [RFC7675].

In case of unstable network conditions, the ICE restart HTTP PATCH requests and responses might be received out of order. In order to mitigate this scenario, when the client performs an ICE restart, it MUST discard any previous ICE username and passwords fragments and ignore any further HTTP PATCH response received from a pending HTTP PATCH request. WHEP players MUST apply only the ICE information received in the response to the last sent request. If there is a

mismatch between the ICE information at the WHEP player and at the WHEP session (because of an out-of-order request), the STUN requests will contain invalid ICE information and will be dropped by the receiving side. If this situation is detected by the WHEP player, it MUST send a new ICE restart request to the WHEP session.

```
PATCH /session/id HTTP/1.1
Host: whep.example.com
If-Match: "*"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 82

a=ice-options:trickle ice2
a=group:BUNDLE 0 1
m=audio 9 UDP/TLS/RTP/SAVPF 111
a=mid:0
a=ice-ufrag:ysXw
a=ice-pwd:vw5LmwG4y/e6dPP/zAP9Gp5k
a=candidate:1387637174 1 udp 2122260223 192.0.2.1 61764 typ host generation 0 ufrag EsAw
network-id 1
a=candidate:3471623853 1 udp 2122194687 198.51.100.2 61765 typ host generation 0 ufrag Es
Aw network-id 2
a=candidate:473322822 1 tcp 1518280447 192.0.2.1 9 typ host tcptype active generation 0 u
frag EsAw network-id 1
a=candidate:2154773085 1 tcp 1518214911 198.51.100.2 9 typ host tcptype active generation
 0 ufrag EsAw network-id 2

HTTP/1.1 200 OK
ETag: "abccd"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 252

a=ice-lite
a=ice-options:trickle ice2
a=group:BUNDLE 0 1
m=audio 9 UDP/TLS/RTP/SAVPF 111
a=mid:0
a=ice-ufrag:289b31b754eaa438
a=ice-pwd:0b66f472495ef0ccac7bda653ab6be49ea13114472a5d10a
a=candidate:1 1 udp 2130706431 198.51.100.1 39132 typ host
a=end-of-candidates
```

Figure 5: Example of an ICE restart request and response

Figure 5 demonstrates a Trickle ICE restart procedure example. The WHEP player sends a PATCH request containing updated ICE information, including a new ufrag and password, along with newly gathered ICE candidates. In response, the WHEP session provides ICE information for the session after the ICE restart, including the updated ufrag and password, as well as the previous ICE candidate.

4.5. WebRTC constraints

To simplify the implementation of WHEP in both players and media servers, WHEP introduces specific restrictions on WebRTC usage. The following subsections will explain these restrictions in detail:

4.5.1. SDP Bundle

Both the WHEP player and the WHEP endpoint SHALL support [RFC9143] and use "max-bundle" policy as defined in [RFC9429]. The WHEP player and the media server MUST support multiplexed media associated with the BUNDLE group as per Section 9 of [RFC9143]. In addition, per [RFC9143] the WHEP player and media server SHALL use RTP/RTCP multiplexing [RFC8858] for all bundled media. In order to reduce the network resources required at the media server, both the WHEP player and WHEP endpoints MUST include the "rtcp-mux-only" attribute in each bundled "m=" sections as per Section 3 of [RFC8858].

4.5.2. Single MediaStream

WHEP only supports a single MediaStream as defined in [RFC8830] and therefore all "m=" sections MUST contain an "msid" attribute with the same value. The MediaStream MUST contain at least one MediaStreamTrack of any media kind and it MUST NOT have two or more than MediaStreamTracks for the same media (audio or video).

4.5.3. Trickle ICE and ICE restarts

The media server SHOULD support full ICE, unless it is connected to the Internet with an IP address that is accessible by each WHEP player that is authorized to use it, in which case it MAY support only ICE lite. The WHEP player MUST implement and use full ICE.

Trickle ICE and ICE restarts support is OPTIONAL for both the WHEP players and media servers as explained in Section 4.4.

4.6. Load balancing and redirections

WHEP endpoints and media servers might not be colocated on the same server, so it is possible to load balance incoming requests to different media servers.

WHEP players SHALL support HTTP redirections as per Section 15.4 of [RFC9110]. In order to avoid POST requests to be redirected as GET requests, status codes 301 and 302 MUST NOT be used and the preferred method for performing load balancing is via the "307 Temporary Redirect" response status code as described in Section 15.4.8 of [RFC9110]. Redirections are not required to be supported for the PATCH and DELETE requests.

In case of high load, the WHEP endpoints MAY return a "503 Service Unavailable" response indicating that the server is currently unable to handle the request due to a temporary overload or scheduled maintenance as described in Section 15.6.4 of [RFC9110], which will likely be alleviated after some delay. The WHEP endpoint might send a Retry-After header field indicating the minimum time that the user agent ought to wait before making a follow-up request as described in Section 10.2.3 of [RFC9110].

4.7. STUN/TURN server configuration

The WHEP Endpoint MAY return STUN/TURN server configuration URLs and credentials usable by the client in the "201 Created" response to the HTTP POST request to the WHEP Endpoint URL.

Each STUN/TURN server will be returned using the "Link" header field [RFC8288] with a "rel" attribute value of "ice-server" as specified in [I-D.draft-ietf-wish-whip]

It might be also possible to configure the STUN/TURN server URLs with long-term credentials provided by either the broadcasting service or an external TURN provider on the WHEP player, overriding the values provided by the WHEP Endpoint.

4.8. Authentication and authorization

All WHEP endpoints, sessions and clients MUST support HTTP Authentication as per Section 11 of [RFC9110] and in order to ensure interoperability, bearer token authentication as defined in the next section MUST be supported by all WHEP entities. However this does not preclude the support of additional HTTP authentication schemes as defined in Section 11.6 of [RFC9110].

4.8.1. Bearer token authentication

WHEP endpoints and sessions MAY require the HTTP request to be authenticated using an HTTP Authorization header field with a Bearer token as specified in Section 2.1 of [RFC6750]. WHEP players MUST implement this authentication and authorization mechanism and send the HTTP Authorization header field in all HTTP requests sent to either the WHEP endpoint or session except the preflight OPTIONS requests for CORS.

The nature, syntax, and semantics of the bearer token, as well as how to distribute it to the client, is outside the scope of this document. Some examples of the kind of tokens that could be used are, but are not limited to, JWT tokens as per [RFC6750] and [RFC8725] or a shared secret stored on a database. The tokens are typically made available to the end user alongside the WHEP endpoint URL and configured on the WHEP players.

WHEP endpoints and sessions could perform the authentication and authorization by encoding an authentication token within the URLs for the WHEP endpoints or sessions instead. In case the WHEP player is not configured to use a bearer token, the HTTP Authorization header field MUST NOT be sent in any request.

4.9. Protocol extensions

In order to support future extensions to be defined for the WHEP protocol, a common procedure for registering and announcing the new extensions is defined.

Protocol extensions supported by the WHEP server MUST be advertised to the WHEP player in the "201 Created" response to the initial HTTP POST request sent to the WHEP Endpoint. The WHEP Endpoint MUST return one "Link" header field for each extension that it supports, with the extension "rel" attribute value containing the extension URN and the URL for the HTTP resource that will be available for receiving requests related to that extension.

Protocol extensions are optional for both WHEP players and WHEP Endpoints and sessions. WHEP players MUST ignore any Link attribute with an unknown "rel" attribute value and WHEP Endpoints and sessions MUST NOT require the usage of any of the extensions.

Each protocol extension MUST register a unique "rel" attribute value at IANA starting with the prefix: "urn:ietf:params:whep:ext" as specified in Section 6.2.

For example, considering a potential extension of server-to-client communication using server-sent events as specified in https://html.spec.whatwg.org/multipage/server-sent-events.html#server-sent-events, the URL for connecting to the server-sent event resource for the ingested stream could be returned in the initial HTTP "201 Created" response with a "Link" header field and a "rel" attribute of "urn:ietf:params:whep:ext:example:server-sent-events" (this document does not specify such an extension, and uses it only as an example).

In this theoretical case, the "201 Created" response to the HTTP POST request would look like:

```
HTTP/1.1 201 Created
Content-Type: application/sdp
Location: https://whep.example.com/session/id
Link: <https://whep.example.com/session/id/sse>;
      rel="urn:ietf:params:whep:ext:example:server-sent-events"
```

Figure 6: Example of a WHEP protocol extension

Figure 6 shows an example of a WHEP protocol extension supported by the WHEP session, as indicated in the Link header of the 201 Created response.

5. Security Considerations

This document specifies a new protocol on top of HTTP and WebRTC, thus, security protocols and considerations from related specifications apply to the WHEP specification. These include:

* WebRTC security considerations: [RFC8826]. HTTPS SHALL be used in order to preserve the WebRTC security model.

* Transport Layer Security (TLS): [RFC8446] and [RFC9147].

* HTTP security: Section 11 of [RFC9112] and Section 17 of [RFC9110].

* URI security: Section 7 of [RFC3986].

On top of that, the WHEP protocol exposes a thin new attack surface specific of the REST API methods used within it:

* HTTP POST flooding and resource exhaustion: It would be possible for an attacker in possession of authentication credentials valid for watching a WHEP stream to make multiple HTTP POST to the WHEP endpoint. This will force the WHEP endpoint to process the

incoming SDP and allocate resources for being able to setup the DTLS/ICE connection. While the malicious client does not need to initiate the DTLS/ICE connection at all, the WHEP session will have to wait for the DTLS/ICE connection timeout in order to release the associated resources. If the connection rate is high enough, this could lead to resource exhaustion on the servers handling the requests and it will not be able to process legitimate incoming ingests. In order to prevent this scenario, WHEP endpoints SHOULD implement a rate limit and avalanche control mechanism for incoming initial HTTP POST requests.

* Insecure direct object references (IDOR) on the WHEP session locations: If the URLs returned by the WHEP endpoint for the WHEP sessions location are easy to guess, it would be possible for an attacker to send multiple HTTP DELETE requests and terminate all the WHEP sessions currently running. In order to prevent this scenario, WHEP endpoints SHOULD generate URLs with enough randomness, using a cryptographically secure pseudorandom number generator following the best practices in Randomness Requirements for Security [RFC4086], and implement a rate limit and avalanche control mechanism for HTTP DELETE requests. The security considerations for Universally Unique IDentifier (UUID) [RFC9562], Section 6 are applicable for generating the WHEP sessions location URL.

* HTTP PATCH flooding: Similar to the HTTP POST flooding, a malicious client could also create a resource exhaustion by sending multiple HTTP PATCH request to the WHEP session, although the WHEP sessions can limit the impact by not allocating new ICE candidates and reusing the existing ICE candidates when doing ICE restarts. In order to prevent this scenario, WHEP endpoints SHOULD implement a rate limit and avalanche control mechanism for incoming HTTP PATCH requests.

6. IANA Considerations

This specification adds a registry for URN sub-namespaces for WHEP protocol extensions.

6.1. Registration of WHEP URN Sub-namespace and WHEP registries

IANA is asked to add an entry to the "IETF URN Sub-namespace for Registered Protocol Parameter Identifiers" registry and create a sub-namespace for the Registered Parameter Identifier as per [RFC3553]: "urn:ietf:params:whep".

To manage this sub-namespace, IANA is asked to create the "WebRTC-HTTP egress protocol (WHEP) URNs" and "WebRTC-HTTP egress protocol (WHEP) extension URNs".

6.1.1. WebRTC-HTTP egress protocol (WHEP) URNs registry

The "WebRTC-HTTP egress protocol (WHEP) URNs" registry is used to manage entries within the "urn:ietf:params:whep" namespace. The registry descriptions is as follows:

* Registry group: WebRTC-HTTP egress protocol (WHEP)
* Registry name: WebRTC-HTTP egress protocol (WHEP) URNs
* Specification: this document (RFC TBD)
* Registration procedure: Specification Required
* Field names: URI, description, change controller, reference and IANA registry reference

The registry contains a single initial value:

* URI: urn:ietf:params:whep:ext
* Description: WebRTC-HTTP egress protocol (WHEP) extension URNs
* Change Controller: IETF
* Reference: this document (RFC TBD) Section Section 6.1.2
* IANA registry reference: WebRTC-HTTP egress protocol (WHEP) extension URNs registry.

6.1.2. WebRTC-HTTP egress protocol (WHEP) extension URNs registry

The "WebRTC-HTTP egress protocol (WHEP) Extension URNs" is used to manage entries within the "urn:ietf:params:whep:ext" namespace. The registry descriptions is as follows:

* Registry group: WebRTC-HTTP egress protocol (WHEP)
* Registry name: WebRTC-HTTP egress protocol (WHEP) Extension URNs
* Specification: this document (RFC TBD)
* Registration procedure: Specification Required

* Field names: URI, description, change controller, reference and IANA registry reference

6.2. URN Sub-namespace for WHEP

WHEP endpoint utilizes URNs to identify the supported WHEP protocol extensions on the "rel" attribute of the Link header as defined in Section 4.9.

This section creates and registers an IETF URN Sub-namespace for use in the WHEP specifications and future extensions.

6.2.1. Specification Template

Namespace ID:

* The Namespace ID "whep" has been assigned.

Registration Information:

* Version: 1

* Date: TBD

Declared registrant of the namespace:

* Registering organization: The Internet Engineering Task Force.

* Designated contact: A designated expert will monitor the WHEP public mailing list, "wish@ietf.org".

Declaration of Syntactic Structure:

* The Namespace Specific String (NSS) of all URNs that use the "whep" Namespace ID shall have the following structure: urn:ietf:params:whep:{type}:{name}:{other}.

* The keywords have the following meaning:

  - type: The entity type. This specification only defines the "ext" type.

  - name: A required ASCII string that conforms to the URN syntax requirements (see [RFC8141]) and defines a major namespace of a WHEP protocol extension. The value MAY also be an industry name or organization name.

- other: Any ASCII string that conforms to the URN syntax requirements (see [RFC8141]) and defines the sub-namespace (which MAY be further broken down in namespaces delimited by colons) as needed to uniquely identify an WHEP protocol extension.

Relevant Ancillary Documentation:

* None

Identifier Uniqueness Considerations:

* The designated contact shall be responsible for reviewing and enforcing uniqueness.

Identifier Persistence Considerations:

* Once a name has been allocated, it MUST NOT be reallocated for a different purpose.

* The rules provided for assignments of values within a sub-namespace MUST be constructed so that the meanings of values cannot change.

* This registration mechanism is not appropriate for naming values whose meanings may change over time.

Process of Identifier Assignment:

* Namespace with type "ext" (e.g., "urn:ietf:params:whep:ext") is reserved for IETF-approved WHEP specifications.

Process of Identifier Resolution:

* None specified.

Rules for Lexical Equivalence:

* No special considerations; the rules for lexical equivalence specified in [RFC8141] apply.

Conformance with URN Syntax:

* No special considerations.

Validation Mechanism:

* None specified.

Scope:

* Global.

6.3. Registering WHEP Protocol Extensions URNs

This section defines the process for registering new WHEP protocol extensions URNs with IANA in the "WebRTC-HTTP egress protocol (WHEP) extension URNs" registry (see Section 6.2).

A WHEP Protocol Extension URNs is used as a value in the "rel" attribute of the Link header as defined in Section 4.9 for the purpose of signaling the WHEP protocol extensions supported by the WHEP endpoints.

WHEP Protocol Extensions URNs have an "ext" type as defined in Section 6.2.

6.3.1. Registration Procedure

The IETF has created a mailing list, "wish@ietf.org", which can be used for public discussion of WHEP protocol extensions proposals prior to registration. Use of the mailing list is strongly encouraged. The IESG has appointed a designated expert as per [RFC8126] who will monitor the wish@ietf.org mailing list and review registrations.

Registration of new "ext" type URNs (in the namespace "urn:ietf:params:whep:ext") belonging to a WHEP Protocol Extension MUST be documented in a permanent and readily available public specification, in sufficient detail so that interoperability between independent implementations is possible and reviewed by the designated expert as per Section 4.6 of [RFC8126]. An Standards Track RFC is REQUIRED for the registration of new value data types that modify existing properties. An Standards Track RFC is also REQUIRED for registration of WHEP Protocol Extensions URNs that modify WHEP Protocol Extensions previously documented in an existing RFC.

The registration procedure begins when a completed registration template, defined in the sections below, is sent to iana@iana.org. Decisions made by the designated expert can be appealed to an Applications and Real Time (ART) Area Director, then to the IESG. The normal appeals procedure described in [BCP9] is to be followed.

Once the registration procedure concludes successfully, IANA creates or modifies the corresponding record in the WHEP Protocol Extension registry.

An RFC specifying one or more new WHEP Protocol Extension URNs MUST include the completed registration templates, which MAY be expanded with additional information. These completed templates are intended to go in the body of the document, not in the IANA Considerations section. The RFC MUST include the syntax and semantics of any extension-specific attributes that may be provided in a Link header field advertising the extension.

6.3.2. Guidance for Designated Experts

The Designated Expert (DE) is expected to ascertain the existence of suitable documentation (a specification) as described in [RFC8126] and to verify that the document is permanently and publicly available.

The DE is also expected to check the clarity of purpose and use of the requested registration.

Additionally, the DE must verify that any request for one of these registrations has been made available for review and comment by posting the request to the WebRTC Ingest Signaling over HTTPS (wish) Working Group mailing list.

Specifications should be documented in an Internet-Draft. Lastly, the DE must ensure that any other request for a code point does not conflict with work that is active in or already published by the IETF.

6.3.3. WHEP Protocol Extension Registration Template

A WHEP Protocol Extension URNs is defined by completing the following template:

* URN: A unique URN for the WHEP Protocol Extension.

* Reference: A formal reference to the publicly available specification

* Description: A brief description of the function of the extension, in a short paragraph or two

* Contact information: Contact information for the organization or person making the registration

7. Acknowledgements

The authors wish to thank Lorenzo Miniero, Juliusz Chroboczek, Adam Roach, Nils Ohlmeier, Christer Holmberg, Cameron Elliott, Gustavo Garcia, Jonas Birme, Sandro Gauci, Christer Holmberg and everyone else in the WebRTC community that have provided comments, feedback, text and improvement proposals on the document and contributed early implementations of the spec.

8. References

8.1. Normative References

[FETCH] WHATWG, "Fetch - Living Standard", n.d., <https://fetch.spec.whatwg.org>.

[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>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <https://www.rfc-editor.org/rfc/rfc3264>.

[RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, DOI 10.17487/RFC3553, June 2003, <https://www.rfc-editor.org/rfc/rfc3553>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/rfc/rfc4086>.

[RFC5789] Dusseault, L. and J. Snell, "PATCH Method for HTTP", RFC 5789, DOI 10.17487/RFC5789, March 2010, <https://www.rfc-editor.org/rfc/rfc5789>.

[RFC6585] Nottingham, M. and R. Fielding, "Additional HTTP Status Codes", RFC 6585, DOI 10.17487/RFC6585, April 2012, <https://www.rfc-editor.org/rfc/rfc6585>.

[RFC6750] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, DOI 10.17487/RFC6750, October 2012, <https://www.rfc-editor.org/rfc/rfc6750>.

[RFC7675] Perumal, M., Wing, D., Ravindranath, R., Reddy, T., and M. Thomson, "Session Traversal Utilities for NAT (STUN) Usage for Consent Freshness", RFC 7675, DOI 10.17487/RFC7675, October 2015, <https://www.rfc-editor.org/rfc/rfc7675>.

[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>.

[RFC8288] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/rfc/rfc8288>.

[RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <https://www.rfc-editor.org/rfc/rfc8445>.

[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/rfc/rfc8446>.

[RFC8725] Sheffer, Y., Hardt, D., and M. Jones, "JSON Web Token Best Current Practices", BCP 225, RFC 8725, DOI 10.17487/RFC8725, February 2020, <https://www.rfc-editor.org/rfc/rfc8725>.

[RFC8826] Rescorla, E., "Security Considerations for WebRTC", RFC 8826, DOI 10.17487/RFC8826, January 2021, <https://www.rfc-editor.org/rfc/rfc8826>.

[RFC8830] Alvestrand, H., "WebRTC MediaStream Identification in the Session Description Protocol", RFC 8830, DOI 10.17487/RFC8830, January 2021, <https://www.rfc-editor.org/rfc/rfc8830>.

[RFC8838] Ivov, E., Uberti, J., and P. Saint-Andre, "Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol", RFC 8838, DOI 10.17487/RFC8838, January 2021, <https://www.rfc-editor.org/rfc/rfc8838>.

[RFC8839] Petit-Huguenin, M., Nandakumar, S., Holmberg, C., Ker辰nen, A., and R. Shpount, "Session Description Protocol (SDP) Offer/Answer Procedures for Interactive Connectivity Establishment (ICE)", RFC 8839, DOI 10.17487/RFC8839, January 2021, <https://www.rfc-editor.org/rfc/rfc8839>.

[RFC8840] Ivov, E., Stach, T., Marocco, E., and C. Holmberg, "A Session Initiation Protocol (SIP) Usage for Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (Trickle ICE)", RFC 8840, DOI 10.17487/RFC8840, January 2021, <https://www.rfc-editor.org/rfc/rfc8840>.

[RFC8858] Holmberg, C., "Indicating Exclusive Support of RTP and RTP Control Protocol (RTCP) Multiplexing Using the Session Description Protocol (SDP)", RFC 8858, DOI 10.17487/RFC8858, January 2021, <https://www.rfc-editor.org/rfc/rfc8858>.

[RFC8863] Holmberg, C. and J. Uberti, "Interactive Connectivity Establishment Patiently Awaiting Connectivity (ICE PAC)", RFC 8863, DOI 10.17487/RFC8863, January 2021, <https://www.rfc-editor.org/rfc/rfc8863>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[RFC9112] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP/1.1", STD 99, RFC 9112, DOI 10.17487/RFC9112, June 2022, <https://www.rfc-editor.org/rfc/rfc9112>.

[RFC9143] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", RFC 9143, DOI 10.17487/RFC9143, February 2022, <https://www.rfc-editor.org/rfc/rfc9143>.

[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/rfc/rfc9147>.

[RFC9429] Uberti, J., Jennings, C., and E. Rescorla, Ed., "JavaScript Session Establishment Protocol (JSEP)", RFC 9429, DOI 10.17487/RFC9429, April 2024, <https://www.rfc-editor.org/rfc/rfc9429>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/rfc/rfc9562>.

[SCTE35] ANSI, "Digital Program Insertion Cueing Message", n.d., <https://account.scte.org/standards/library/catalog/scte-35-digital-program-insertion-cueing-message>.

[W3C.REC-ldp-20150226] Malhotra, A., Ed., Arwe, J., Ed., and S. Speicher, Ed., "Linked Data Platform 1.0", W3C REC REC-ldp-20150226, W3C REC-ldp-20150226, 26 February 2015, <https://www.w3.org/TR/2015/REC-ldp-20150226/>.

8.2. Informative References

[BCP56] Best Current Practice 56, <https://www.rfc-editor.org/info/bcp56>. At the time of writing, this BCP comprises the following:

Nottingham, M., "Building Protocols with HTTP", BCP 56, RFC 9205, DOI 10.17487/RFC9205, June 2022, <https://www.rfc-editor.org/info/rfc9205>.

[BCP9] Best Current Practice 9, <https://www.rfc-editor.org/info/bcp9>. At the time of writing, this BCP comprises the following:

Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, DOI 10.17487/RFC2026, October 1996, <https://www.rfc-editor.org/info/rfc2026>.

Dusseault, L. and R. Sparks, "Guidance on Interoperation and Implementation Reports for Advancement to Draft Standard", BCP 9, RFC 5657, DOI 10.17487/RFC5657, September 2009, <https://www.rfc-editor.org/info/rfc5657>.

Housley, R., Crocker, D., and E. Burger, "Reducing the Standards Track to Two Maturity Levels", BCP 9, RFC 6410, DOI 10.17487/RFC6410, October 2011, <https://www.rfc-editor.org/info/rfc6410>.

Resnick, P., "Retirement of the "Internet Official Protocol Standards" Summary Document", BCP 9, RFC 7100, DOI 10.17487/RFC7100, December 2013, <https://www.rfc-editor.org/info/rfc7100>.

Kolkman, O., Bradner, S., and S. Turner, "Characterization of Proposed Standards", BCP 9, RFC 7127, DOI 10.17487/RFC7127, January 2014, <https://www.rfc-editor.org/info/rfc7127>.

Dawkins, S., "Increasing the Number of Area Directors in an IETF Area", BCP 9, RFC 7475, DOI 10.17487/RFC7475, March 2015, <https://www.rfc-editor.org/info/rfc7475>.

Halpern, J., Ed. and E. Rescorla, Ed., "IETF Stream Documents Require IETF Rough Consensus", BCP 9, RFC 8789, DOI 10.17487/RFC8789, June 2020, <https://www.rfc-editor.org/info/rfc8789>.

Rosen, B., "Responsibility Change for the RFC Series", BCP 9, RFC 9282, DOI 10.17487/RFC9282, June 2022, <https://www.rfc-editor.org/info/rfc9282>.

[I-D.draft-ietf-wish-whip]
Murillo, S. G. and A. Gouaillard, "WebRTC-HTTP ingestion protocol (WHIP)", Work in Progress, Internet-Draft, draft-ietf-wish-whip-16, 21 August 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-wish-whip-16>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/rfc/rfc3261>.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <https://www.rfc-editor.org/rfc/rfc6120>.

[RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <https://www.rfc-editor.org/rfc/rfc7826>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017, <https://www.rfc-editor.org/rfc/rfc8141>.

[RFC9457] Nottingham, M., Wilde, E., and S. Dalal, "Problem Details for HTTP APIs", RFC 9457, DOI 10.17487/RFC9457, July 2023, <https://www.rfc-editor.org/rfc/rfc9457>.

Authors' Addresses

Sergio Garcia Murillo
Millicast
Email: sergio.garcia.murillo@cosmosoftware.io

Cheng Chen
ByteDance
Email: webrtc@bytedance.com

Dan Jenkins (editor)
Everycast Labs Ltd
Email: dan@everycastlabs.uk