Workgroup: Network Working Group
Internet-Draft: draft-murillo-whep-02
Published: 29 March 2023
Intended Status: Informational
Expires: 30 September 2023
Authors: S. Murillo C. Chen
Millicast ByteDance

WebRTC-HTTP Egress Protocol (WHEP)

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 30 September 2023.

Table of Contents

1. Introduction

The IETF RTCWEB working group standardized JSEP ([RFC8829]), 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] as well as 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. This flexibility has allowed the implementation of a wide range of services. However, those services are typically standalone silos which don't require interoperability with other services or leverage the existence of tools that can communicate with them.

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, is not compatible with the SDP offer/answer model [RFC3264].

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 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.

- WHEP Player: WebRTC media player that acts as a client of the WHEP protocol by receiving and decoding the media from a remote Media Server.

- WHEP Endpoint: Egress server receiving the initial WHEP request.

- WHEP Endpoint URL: URL of the WHEP endpoint that will create the WHEP resource.

- Media Server: WebRTC Media Server or consumer that establishes the media session with the WHEP player and delivers the media to it.

- WHEP Resource: Allocated resource by the WHEP endpoint for an ongoing egress session that the WHEP player can send requests for altering the session (ICE operations or termination, for example).

- WHEP Resource URL: URL allocated to a specific media session by the WHEP endpoint which can be used to perform operations such as terminating the session or ICE restarts.

3. Overview

The WebRTC-HTTP Egress Protocol (WHEP) uses an HTTP POST request to perform a single-shot SDP offer/answer so an ICE/DTLS session can be established between the WHEP Player and the streaming service endpoint (Media Server).

Once the ICE/DTLS session is set up, the media will flow unidirectionally from Media Server to the WHEP Player. In order to reduce complexity, no SDP renegotiation is supported, so no "m=" sections can be added once the initial SDP offer/answer over HTTP is completed.

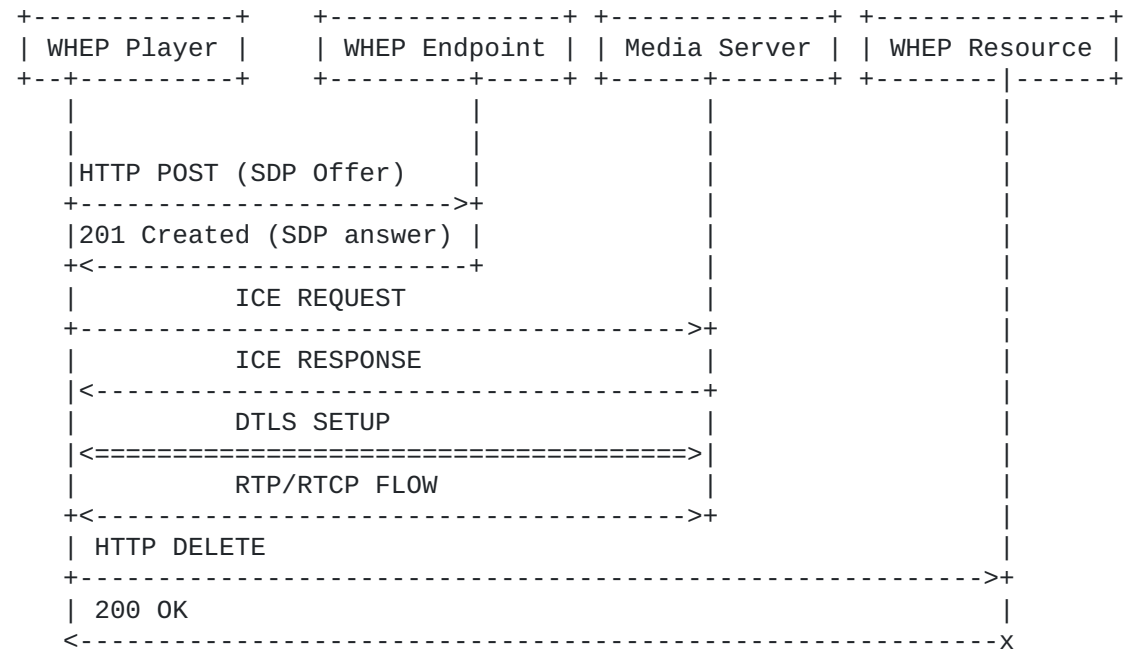

Figure 1: WHEP session setup and teardown

4. Protocol Operation

In order to set up a streaming session, the WHEP Player will generate an SDP offer according to the JSEP rules and perform an HTTP POST request to the configured WHEP Endpoint URL.

The HTTP POST request will have a content type of "application/sdp" and contain the SDP offer as the body. The WHEP Endpoint will generate an SDP answer 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 resource.

The SDP offer **SHOULD** use the "recvonly" attribute and the SDP answer **MUST** use "sendonly" the attribute.

```
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=msid-semantic: WMS
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=ice-options:trickle
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58
a=setup:actpass
a=mid:0
a=bundle-only
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=recvonly
a=rtcp-mux
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
m=video 9 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=ice-options:trickle
a=fingerprint:sha-256 DA:7B:57:DC:28:CE:04:4F:31:79:85:C4:31:67:EB:27:58
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-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/resource/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=msid-semantic: WMS *
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
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-rsize
a=rtpmap:111 opus/48000/2
a=fmtp:111 minptime=10;useinbandfec=1
a=msid:- d46fb922-d52a-4e9c-aa87-444eadc1521b
m=video 9 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
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-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: HTTP POST and PATCH doing SDP O/A example

The WHEP Resource 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 Resource **SHALL** return a "409 Conflict" response to the POST request issued by the WHEP Client 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 Resource with exponential backoff period with an initial value of the Retry-After header value in the "409 Conflict" response.

Once a session is set up, ICE consent freshness [RFC7675] **SHALL** be used to detect abrupt disconnection and DTLS teardown for session termination by either side.

To explicitly terminate a session, the WHEP Player **MUST** perform an HTTP DELETE request to the resource URL returned in the Location header field of the initial HTTP POST. Upon receiving the HTTP DELETE request, the WHEP resource 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 [RFC7675] section 5.2 for immediate revocation of consent.

The WHEP Endpoints **MUST** return an "405 Method Not Allowed" response for any HTTP GET, HEAD or PUT requests on the endpoint URL in order to reserve its usage for future versions of this protocol specification.

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

The WHEP Resources **MUST** return an"405 Method Not Allowed" for any HTTP GET, HEAD, POST or PUT requests on the resource URL in order to reserve its usage for future versions of this protocol specification.

4.1. ICE and NAT support

The SDP provided by the WHEP Player **MAY** be sent after the full ICE gathering is complete with the full list of ICE candidates, or it **MAY** only contain local candidates (or even an empty list of candidates) as per [RFC8863].

In order to simplify the protocol, there is no support for exchanging gathered trickle candidates from Media Server ICE candidates once the SDP answer is 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. The Media Server **MAY** use ICE lite, while the WHEP player **MUST** implement full ICE.

Trickle ICE and ICE restart support is **OPTIONAL** for a WHEP resource.

If the WHEP resource supports either Trickle ICE or ICE restarts, the WHEP player **MUST** include an "Accept-Patch" header with a mime type value of "application/trickle-ice-sdpfrag" in the "201 Created" of the POST request that creates the WHEP resource as per [RFC5789] section 3.1.

If the WHEP resource supports either Trickle ICE or ICE restarts, but not both, it **MUST** return a "405 Not Implemented" response for the HTTP PATCH requests that are not supported.

If the WHEP resource does not support the PATCH method for any purpose, it **MUST** return a "501 Not Implemented" response, as described in [RFC9110] section 6.6.2.

As the HTTP PATCH request sent by a WHEP player may be received out-of-order by the WHEP resource, the WHEP resource **MUST** generate a unique strong entity-tag identifying the ICE session as per [RFC9110] section 2.3. 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. It **MUST** also be returned in the "200 OK" of any PATCH request that triggers an ICE restart. Note that including the ETag in the original "201 Created" response is only **REQUIRED** if the WHEP resource supports ICE restarts and **OPTIONAL** otherwise.

A WHEP player sending a PATCH request for performing trickle ICE **MUST** include an "If-Match" header field with the latest known entity-tag as per [RFC9110] section 3.1. When the PATCH request is received by the WHEP resource, it **MUST** compare the indicated entity-tag value with the current entity-tag of the resource as per [RFC9110] section 3.1 and return a "412 Precondition Failed" response if they do not match.

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.

A WHEP resource receiving a PATCH request with new ICE candidates, but which does not perform an ICE restart, **MUST** return a "204 No Content" response without body. If the Media Server does not support a candidate transport or is not able to resolve the connection address, it **MUST** accept the HTTP request with the "204 No Content" response and silently discard the candidate.

```
PATCH /resource/id HTTP/1.1
Host: whep.example.com
If-Match: "38sdf4fdsf54:EsAw"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 548

a=ice-ufrag:EsAw
a=ice-pwd:P2uYro0UCOQ4zxjKXaWCBui1
m=audio RTP/AVP 0
a=mid:0
a=candidate:1387637174 1 udp 2122260223 192.0.2.1 61764 typ host generat
a=candidate:3471623853 1 udp 2122194687 198.51.100.1 61765 typ host gene
a=candidate:473322822 1 tcp 1518280447 192.0.2.1 9 typ host tcptype acti
a=candidate:2154773085 1 tcp 1518214911 198.51.100.2 9 typ host tcptype
a=end-of-candidates

HTTP/1.1 204 No Content
```

Figure 3: Trickle ICE request

A WHEP Player sending a PATCH request for performing ICE restart **MUST** contain an "If-Match" header field with a field-value "*" as per [RFC9110] section 3.1.

If the HTTP PATCH request results in an ICE restart, the WHEP resource **SHALL** return a "200 OK" with an "application/trickle-ice-sdpfrag" body containing the new ICE username fragment and password and OPTIONALLY a new set of ICE candidates for the WHIP client . 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.

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

```
PATCH /resource/id HTTP/1.1
Host: whep.example.com
If-Match: "*"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 54

a=ice-ufrag:ysXw
a=ice-pwd:vw5LmwG4y/e6dPP/zAP9Gp5k

HTTP/1.1 200 OK
ETag: "289b31b754eaa438:ysXw"
Content-Type: application/trickle-ice-sdpfrag
Content-Length: 102

a=ice-lite
a=ice-ufrag:289b31b754eaa438
a=ice-pwd:0b66f472495ef0ccac7bda653ab6be49ea13114472a5d10a
```

Figure 4: ICE restart request

Because the WHEP Player needs to know the entity-tag associated with the ICE session in order to send new ICE candidates, it **MUST** buffer any gathered candidates before it receives the HTTP response to the initial POST request or the PATCH request with the new entity-tag value. Once it knows the entity-tag value, the WHEP Player **SHOULD** send a single aggregated HTTP PATCH request with all the ICE candidates it has buffered so far.

In case of unstable network conditions, the ICE restart HTTP PATCH requests and responses might be received out of order. n 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 client and at the server (because of an out-of-order request), the STUN requests will contain invalid ICE information and will be rejected by the server. When this situation is detected by the WHEP Player, it **SHOULD** send a new ICE restart request to the server.

4.2. WebRTC constraints

In the specific case of media consumption from a streaming service, some assumptions can be made about the server-side which simplifies the WebRTC compliance burden, as detailed in WebRTC-gateway document [I-D.draft-ietf-rtcweb-gateways].

In order to reduce the complexity of implementing WHEP in both players and Media Servers, WHEP imposes the following restrictions regarding WebRTC usage:

Both the WHEP Player and the WHEP Endpoint **SHALL** use SDP bundle [RFC9143]. Each "m=" section **MUST** be part of a single BUNDLE group. Hence, when a WHEP Playersends an SDP offer, it **MUST** include a "bundle-only" attribute in each bundled "m=" section. The WHEP player and the Media Server **MUST** support multiplexed media associated with the BUNDLE group as per [RFC9143] section 9. In

addition, per [RFC9143] the WHEP Player and Media Server will use RTP/RTCP multiplexing for all bundled media. The WHEP Player and Media Server **SHOULD** include the "rtcp-mux-only" attribute in each bundled "m=" section as per [RFC8858].

Trickle ICE and ICE restarts support is **OPTIONAL** for both the WHEP Players and Media Servers as explained in section 4.1.

4.3. Load balancing and redirections

WHEP Endpoints and Media Servers might not be co-located on the same server, so it is possible to load balance incoming requests to different Media Servers. WHEP Players **SHALL** support HTTP redirection via the "307 Temporary Redirect" response as described in [RFC9110] section 6.4.7. The WHEP Resource URL **MUST** be a final one, and redirections are not required to be supported for the PATCH and DELETE requests sent to it.

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, 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.

4.4. 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.

4.5. Authentication and authorization

WHEP Endpoints and Resources **MAY** require the HTTP request to be authenticated using an HTTP Authorization header field with a Bearer token as specified in [RFC6750] section 2.1. 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 resource 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.

WHEP Endpoints and Resources could perform the authentication and authorization by encoding an authentication token within the URLs for the WHEP Endpoints or Resources 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.

4.6. 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, with the extension "rel" type attribute and the URI 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 Resources. WHEP Players **MUST** ignore any Link attribute with an unknown "rel" attribute value and WHEP Endpoints and Resources **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.

In the first version of the WHEP specification, two optional extensions are defined: the Server Sent Events and the Video Layer Selection.

4.6.1. Server Sent Events extension

This optional extesion provides support for server-to-client communication using WHATWG server sent events protocol as specified in https://html.spec.whatwg.org/multipage/server-sent-events.html#server-sent-events. When supported by the WHEP resource, a "Link" header field with a "rel" attribute of "urn:ietf:params:whep:ext:sse" **MUST** be returned in the initial HTTP "201 Created" response, with the Url of the Server Sent Events REST API entrypoint. The "Link" header field **MAY** also contain an "events" attribute with a coma separated list of supported event types.

```
HTTP/1.1 201 Created
Content-Type: application/sdp
Location: https://whep.example.org/resource/213786HF
Link: <https://whep.ietf.org/resource/213786HF/sse>;
      rel="urn:ietf:params:whep:ext:core:server-sent-events"
      events="active,inactive,layers,viewercount"
```

Figure 5: HTTP 201 response example containing the Server Sent Events extension

If the extension is also supported by the WHEP player, it **MAY** send a POST request to the Server Sent Events REST API entrypoint to create

a server-to-client event stream using WHATWG server sent events protocol. The POST request **MAY** contain an "application/json" body with an JSON array indicating the subset of the event list announced by the WHEP Resource on the "events" atribute which COULD be sent by the server using the server-to-client communication channel. The WHEP Endpoint will return a "201 Created" response with a Location header field pointing to the newly created server-to-client event stream.

```
POST /resource/213786HF/sse HTTP/1.1
Host: whep.example.com
Content-Type: application/sjon

["active","inactive","layers","viewercount"]

HTTP/1.1 201 Created
Location: https://whep.example.org/resource/213786HF/sse/event-stream
```

Figure 6: HTTP POST request to create a server-to-client event stream

Once the server-to-client communication channel has been created the WHEP player can perform a long pull using the Url returned on the location header as expecified in the WHATWG server sent events protocol.

When an event is generated, the WHEP Resource **MUST** check for each event stream if the type is on the list provided by the WHEP player when the event stream was created, and if so enque it for delivering when an active long pull request is available.

The events types supported by this specification are the following:

- *active: indicating that there is an active publication ongoing for this resource.
- *inactive: indicating that there is no active publication ongoing for this resource.
- *layers: provides information about the video layers being published for this resource.
- *viewercount: provides the number of viewers currently connected to this resource.

The WHEP resource must indicate the event type in the "event" field and a JSON serialized string in the "data" field of the WHATWG server sent events message. In order to make the processing simpler on the WHEP Player, the WHEP resource **MUST** encode the event data in a single "data" line.

```
event: viewercount
data: {"viewercount":3}
```

Figure 7: Example event

The WHEP Player **MAY** destroy the event stream at anytime by sending a HTTP DELETE request to the Url returned on the location header on the created request. The WHEP Resource **MUST** drop any pending queued event and return a "404 Not found" if any further long pull request is received for the event stream.

All the event streams associated with a WHEP Resource **MUST** be destroyed when the WHEP Resource is terminated.

4.6.1.1. active event

The event is sent by the WHEP Resource when an active publication for the WHEP resource, either at the begining of the playback when the resource is created or later during the playback session.

*event name: "active"

*event data: JSON object (TBD)

4.6.1.2. inactive event

The event is sent by the WHEP Resource when an active publication is no longer available. The WHEP Resource **MUST** not send an initial "inactive" event if there is no active publication when the resource is created.

*event name: "active"

*event data: JSON object (TBD)

4.6.1.3. layers event

The event is sent by the WHEP Resource to provide information to the WHEP player about the avialable video layers or renditions to be used in conjuction with the Layer Selection extension defined in Chapter {TBD}.

*event name: "layers"

*event data: JSON object

The WHEP Resource **MAY** send the event periodically or just when the layer information has changed.

The event data JSON object contains the video layer information available for each "m-line" indexed by the "m-line" order in the SDP.

Each value of the JSON object entries will be a JSON object with the following attributes

*active: (Array<Object>) Containing the information of the active simulcast layers.

*inactive: (Array<Object>) Containing the information of the inactive simulcast layers.

*layers: (Array<Object>) Containing the information of the active simulcast, spatials or temporal layers available for layer selection.

Each "active" JSON objet contains the following information:

*id: (String) rid value of the simulcast encoding of the layer

*simulcastIdx: (Number) the simulcast order of the encoding layer.

*bitrate: (Number) the spatial layer id of the encoding layer.

*width: (Number) the current video with of the encoding layer.

*heigth: (Number) the current video height of the encoding layer.

Each "inactive" JSON contains the following information:

*id: (String) rid value of the simulcast encoding of the layer.

*simulcastIdx: (Number) the simulcast order of the encoding layer.

*width: (Number) the current video with of the encoding layer

*heigth: (Number) the current video height of the encoding layer.

Each "layer" JSON contains the following information:

*encodingId: (String) rid value of the simulcast encoding of the layer

*simulcastIdx: (Number) the simulcast order of the encoding layer.

*spatialLayerId: (Number) the spatial layer id of the encoding layer.

*temporalLayerId: (Number) the temporal layer id of the encoding layer.

*bitrate: (Number) the spatial layer id of the encoding layer.

*width: (Number) the current video with of the encoding layer.

*heigth: (Number) the current video height of the encoding layer.

The "layer" object **MUST** containt at least one of the encodingId, spatialLayerId or temporalLayerId attributes, the other attributes are **OPTIONAL**.

```
{
  "0": {
    "active": [
      {
        "id": "1", "simulcastIdx": 1, "bitrate": 538288, width: 640, hei
      },
      {
        "id": "0", "simulcastIdx": 0, "bitrate": 111600, width: 320, hei
      }
    ],
    "inactive": [
      {
        "id": "2", "simulcastIdx": 2
      },
    ],
    "layers": [
      { "encodingId": "1", "simulcastIdx": 1, "spatialLayerId": 0, "temp
      { "encodingId": "1", "simulcastIdx": 1, "spatialLayerId": 0, "temp
      { "encodingId": "0", "simulcastIdx": 0, "spatialLayerId": 0, "temp
      { "encodingId": "0", "simulcastIdx": 0, "spatialLayerId": 0, "temp
    ]
  }
}
```

Figure 8: Example event

4.6.1.4. viewercount event

The event is sent by the WHEP Resource to provide the WHIP Player the information of number of viewers currently connected to this resource.

*event name: "viewercount"

*event data: JSON object containing a "viewercount" attribute with a Number value indicating the number of viewers currently watching the WHIP resource.

The viewer count provided by the WHEP Resource **MAY** be approximate and not updated in real time but periodically to avoid overloading both the event stream and the Media Server.

4.6.2. Video Layer Selection extension

The Layer Selection extensions allows the WHEP Player to control which video layer or rendition is being delivered through the negotiated video MediaStreamTrack. When supported by the WHEP resource, a "Link" header field with a "rel" attribute of "urn:ietf:params:whep:ext:layer" **MUST** be returned in the initial HTTP "201 Created" response, with the Url of the Video Layer Selection REST API entrypoint. If this extension is supported by the WHEP Resource, the Server Sent Events extension **MUST** be supported as well and the "layers" event **MUST** be advertised as well.

```
HTTP/1.1 201 Created
Content-Type: application/sdp
Location: https://whep.example.org/resource/213786HF
Link: <https://whep.ietf.org/resource/213786HF/layer>;
      rel="urn:ietf:params:whep:ext:core:layer"
Link: <https://whep.ietf.org/resource/213786HF/layer>;
      rel="urn:ietf:params:whep:ext:core:server-sent-events"
      events="layers"
```

Figure 9: HTTP 201 response example containing the Video Layer Selection extension

In case that Simulcast or Scalable Video Codecs are supported by the Media Server and used in the active publication to the WHEP Resource, by default, the Media Server will choose one of the available video layers to be sent to the WHEP Player (based on bandwidth estimation or any other business logic). However, the WHEP Player (or the person watching the stream) may decide that it whishes to receive a different one (to preserve bandwidth or to best fit in the UI). In this case the WHEP Player **MAY** send a HTTP POST request to theVideo Layer Selection API entrypoint containing an "application/json" body with an JSON object indicating the information of the video layer that wishes to be received. The WHEP Endpoint will return a "200 OK" if the switch to the new video layer can be performed or an appropiate HTTP error response if not.

The information that can sent on the JSON object in the POST request for doing layer selection is as follows:

- *mediaId: (String) m-line index to apply the layer selection(default: first video m-line)
- *encodingId: (String) rid value of the simulcast encoding of the track (default: automatic selection)
- *spatialLayerId: (Number) The spatial layer id to send to the outgoing stream (default: max layer available)
- *temporalLayerId: (Number) The temporaral layer id to send to the outgoing stream (default: max layer available)
- *maxSpatialLayerId: (Number) Max spatial layer id (default: unlimited)
- *maxTemporalLayerId: (Number) Max temporal layer id (default: unlimited)

The information about the avialable encodings, spatial or temporal layers should be retrieverd from a "layers" event sent by the WHEP Resource using the Server Sent Events extension:

```
POST /resource/213786HF/layer HTTP/1.1
Host: whep.example.com
Content-Type: application/sjon

{mediaId:"0", "encodingId": "hd"}

HTTP/1.1 200 OK
```

If the WHEP Player wishes to return to the default selection performed by the Media Server, it just need to send an empty JSON Object instead:

```
POST /resource/213786HF/layer HTTP/1.1
Host: whep.example.com
Content-Type: application/sjon

{}

HTTP/1.1 200 OK
```

5. Security Considerations

HTTPS **SHALL** be used in order to preserve the WebRTC security model.

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 Registry

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

To manage this sub-namespace, IANA has created the "System for Cross-domain Identity Management (WHEP) Schema URIs" registry, which is used to manage entries within the "urn:ietf:params:whep" namespace. The registry description is as follows:

- *Registry name: WHEP
- *Specification: this document (RFC TBD)
- *Repository: See Section Section 6.2
- *Index value: See Section Section 6.2

6.2. URN Sub-namespace for whep

whep Endpoint utilize URIs to identify the supported whep protocol extensions on the "rel" attribute of the Link header as defined in Section 4.6. 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:

The Internet Engineering Task Force.

Designated contact:

A designated expert will monitor the whep public mailing list, "wish@

Declaration of Syntactic Structure:

The Namespace Specific String (NSS) of all URNs that use the "whep" Na

The keywords have the following meaning:

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

- name: A required US-ASCII string that conforms to the URN syntax req

- other: Any US-ASCII string that conforms to the URN syntax requireme

Relevant Ancillary Documentation:

None

Identifier Uniqueness Considerations:

The designated contact shall be responsible for reviewing and enforcin

Identifier Persistence Considerations:

Once a name has been allocated, it MUST NOT be reallocated for a diffe
The rules provided for assignments of values within a sub-namespace MU
This registration mechanism is not appropriate for naming values whose

Process of Identifier Assignment:

Namespace with type "ext" (e.g., "urn:ietf:params:whep:ext") is reserv

Process of Identifier Resolution:

None specified.

Rules for Lexical Equivalence:

No special considerations; the rules for lexical equivalence specified

Conformance with URN Syntax:

No special considerations.

Validation Mechanism:

None specified.

Scope:

Global.

7. Acknowledgements

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>.

[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>.

[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>.

[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>.

[RFC8829] Uberti, J., Jennings, C., and E. Rescorla, Ed., "JavaScript Session Establishment Protocol (JSEP)", RFC

8829, DOI 10.17487/RFC8829, January 2021, <https://www.rfc-editor.org/rfc/rfc8829>.

[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>.

[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>.

[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

[I-D.draft-ietf-rtcweb-gateways] Alvestrand, H. T. and U. Rauschenbach, "WebRTC Gateways", Work in Progress, Internet-Draft, draft-ietf-rtcweb-gateways-02, 21 January 2016, <https://datatracker.ietf.org/doc/html/draft-ietf-rtcweb-gateways-02>.

[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-07, 13 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-wish-whip-07>.

[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>.

[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>.

Authors' Addresses

Sergio Garcia Murillo
Millicast

Email: sergio.garcia.murillo@cosmosoftware.io

Cheng Chen
ByteDance

Email: webrtc@bytedance.com