W3C WebRTC WG Meeting

July 19, 2022
8 AM - 10 AM

Chairs: Bernard Aboba
Harald Alvestrand
Jan-Ivar Bruaroey

W3C WG IPR Policy

- This group abides by the W3C Patent Policy https://www.w3.org/Consortium/Patent-Policy/
- Only people and companies listed at https://www.w3.org/2004/01/pp-impl/47318/status are allowed to make substantive contributions to the WebRTC specs

Welcome!

- Welcome to the July 2022 interim meeting of the W3C WebRTC WG, at which we will cover:
  - WebRTC-Extensions
  - WebRTC-PC
- Future meetings:
  - TPAC 2022

About this Virtual Meeting

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/July_19_2022
- Link to latest drafts:
  - https://w3c.github.io/mediacapture-main/
  - https://w3c.github.io/mediacapture-extensions/
  - https://w3c.github.io/mediacapture-image/
  - https://w3c.github.io/mediacapture-output/
  - https://w3c.github.io/mediacapture-screen-share/
  - https://w3c.github.io/mediacapture-record/
  - https://w3c.github.io/webrtc-pc/
  - https://w3c.github.io/webrtc-extensions/
  - https://w3c.github.io/webrtc-stats/
  - https://w3c.github.io/mst-content-hint/
  - https://w3c.github.io/webrtc-priority/
  - https://w3c.github.io/webrtc-nv-use-cases/
  - https://github.com/w3c/webrtc-encoded-transform
  - https://github.com/w3c/mediacapture-transform
  - https://github.com/w3c/webrtc-svc
  - https://github.com/w3c/webrtc-ice
- Link to Slides has been published on WG wiki
- Scribe? IRC http://irc.w3.org/ Channel: #webrtc
- The meeting is (still) being recorded. The recording will be public.
- Volunteers for note taking?

W3C Code of Conduct

- This meeting operates under W3C Code of Ethics and Professional Conduct

- We're all passionate about improving WebRTC and the Web, but let's all keep the conversations cordial and professional

Virtual Interim Meeting Tips

This session is (still) being recorded

- Type +q and -q in the Google Meet chat to get into and out of the speaker queue.
- Please use headphones when speaking to avoid echo.
- Please wait for microphone access to be granted before speaking.
- Please state your full name before speaking.
- Poll mechanism may be used to gauge the “sense of the room”.

Understanding Document Status

- Hosting within the W3C repo does ***not*** imply adoption by the WG.
    - WG adoption requires a Call for Adoption (CfA) on the mailing list.
- Editor’s drafts do ***not*** represent WG consensus.
    - WG drafts ***do*** imply consensus, once they’re confirmed by a Call for Consensus (CfC) on the mailing list.
    - Possible to merge PRs that may lack consensus, if a note is attached indicating controversy.

Issues for Discussion Today

- 08:10 - 08:50 AM WebRTC-Extensions
  - Slides (08:10 - 08:30)
  - Discussion (08:30 - 08:50)
- 08:50 - 09:30 WebRTC-PC
  - Slides (08:50 - 09:10)
  - Discussion (09:10 - 9:30)
- 9:30 - 09:40 AM Wrap-up and Next Steps

Time control:

- A warning will be given 2 minutes before time is up.
- Once time has elapsed we will move on to the next item.

WebRTC-Extensions

Start Time: 8:10 AM

End Time: 8:50 AM

For Discussion Today

- WebRTC-Extensions
  - Issue 71: Add SCTP rate control params to RTCPeerConnection constructor (Bernard)
  - Issue 107: maxFramerate probably should not be allowed in addTransceiver/setParameters for audio senders (Bernard)
  - Issue 110: getDataChannels() method on RTCPeerConnection (Florent)

Issue 71: Add SCTP rate control params to RTCPeerConnection constructor

- Use case: Terminal application or “PC in the Cloud”
- Problem:
  - When congestion is encountered, RTO backs off exponentially to a maximum of 60 seconds.
  - Users experiences unacceptable delay when application uses reliable/ordered transport.
- Notes
  - RTOMax, RTOMin and RTOInitial are properties of the SCTP connection and cannot be set on a per-datachannel basis.
  - Recommended values for these parameters are provided in RFC 4960, Section 15:
    - RTO.Initial - 3 seconds, RTO.Min - 1 second, RTO.Max - 60 seconds
- **Recommendation**: Resolve as “won’t fix”
  - Applications can choose unreliable/unordered transport and handle retransmission and/or FEC themselves.
  - Values for RTO.Initial, RTO.min and RTO.max are carefully considered and changes to recommendations are typically documented in an RFC (e.g. RFC 6298 for TCP), not left to the application.

Issue 107: maxFramerate probably should not be allowed in addTransceiver/setParameters for audio senders

- `maxFramerate` was intended as a video-only parameter.
- **Recommendation**: Label as “Ready for PR”
  - Throw an error in `addTransceiver` and `setParameters` if `maxFramerate` is used with `track.kind == ‘audio’`

Issue 110: getDataChannels() method on RTCPeerConnection (Florent)

- We have: `getTransceivers()`, `getSenders()`, `getReceivers()`
- But we don’t have a way to access all the data channels that have been created with the peer connection.
- Proposed solution: Add `getDataChannels()`

- Goals:
  - List only non-closed channels (or we run into the possibility of having an unbounded list)
  - Both created locally and remotely
  - Channels created before first negotiation (and SCTP transport creation) should be listed too
- Recommendation: Label as “Ready for PR”

Discussion (End Time: 8:50 AM)

-

WebRTC-PC

Start Time: 8:50 AM

End Time: 09:30 AM

For Discussion Today

- Issue 2735: webrtc-pc specifies using '~' in a=simulcast, but does not support RFC 7728 (RTP pause)(Philipp)
- Issue 2746: **`Enum RTCIceCredentialType`** with only one value (Florent)
- Issue 2743: SLD/SRD(answer) trips over itself when narrowing simulcast envelope (Jan-Ivar)
- Issue 2751: Intended outcome when modifying direction in have-local-offer (Jan-Ivar)
- Issue 2722: sRD(offer) completely overwrites pre-existing **`transceiver.[[Sender]].[[SendEncodings]]`** (Bernard)
- Issue 2723: The prose around "simulcast envelope" falsely implies that simulcast encodings can never be removed (Bernard)
- Issue 2724: The language around setting a description appears to prohibit renegotiation of RIDs (Jan-Ivar)

Issue 2735: webrtc-pc specifies using '~' in a=simulcast, but does not support RFC 7728 (RTP pause)

- RFC 8853 does not mandate or recommend support for RFC 7728, which is IPR encumbered and is not implemented in any browser.
- SDP can control whether a stream gets sent initially or not by prefixing the rid with a ~ character
- API level problem: this implies setParameters changes SDP, which would require triggering ONN.
  - From Section 6.2: "`setParameters` does not cause SDP renegotiation and can only be used to change what the media stack is sending or receiving within the envelope negotiated by Offer/Answer."
  - Privacy problem: remote can re-enable layers disabled locally
- Recommendation: ignore ~
  - Remove text from §4.4.1.5 about processing it for remote descriptions
  - Currently implemented by Chrome but existing bug, does not support RTP pause

Issue 2746: Enum RTCIceCredentialType with only one value (Florent)

- Currently, the enum has a single possible value, which is the default one in `RTCIceServer`.
- Do we have plans to bring back an oauth or alternate type or should we try to remove this?
  - Firefox appears to be the only browser that has implemented this.
  - `RTCIceCredentialType` “oauth” enum is defined in the Removed Features section of WebRTC-Extensions (Section 13.2)

[Issue 2743](): SLD/SRD (answer) trips over itself when narrowing simulcast envelope (Jan-Ivar)

- #2081 added simulcast rejection in answer (by truncating layers): *"If ... simulcast is not supported or desired, ... description rejects any of the offered layers, ... [OR] update the paused status ... [THEN MODIFY] transceiver.[[Sender]].[[SendEncodings]]"*
- Unfortunately, this runs afoul of language added in #2314: *"If applying description leads to modifying a transceiver transceiver, and transceiver.[[Sender]].[[SendEncodings]] is non-empty, and not equal to the encodings that would result from processing description, the process of applying description fails."*
- When read together, this says SRD(answer) must fail the following if the answer doesn't support simulcast or rejects a layer:
  - `pc.addTransceiver("video", {sendEncodings: [{rid: "a"}, {rid: "b"}]})`

Issue 2743: SLD/SRD (answer) trips over itself when narrowing simulcast envelope (cont'd)

This seems accidental, because:

1. failing defeats the detailed modifications that Clarifying how the simulcast envelope is created. #2081 made.
2. Can simulcast offers renegotiate rids? #2314 say its purpose was to *"not allow remotely initiated RID renegotiation"*.

Remotely initiated = remote offer, not remote answer.

The solution would be to narrow the #2314 language so it doesn't catch #2081.

- Should we label this Issue "Ready for PR"?

Issue 2751: Intended outcome when modifying direction in have-local-offer (Jan-Ivar)

In #2033 we set `tc.direction` in SRD(answer), which was a mistake and is unimplemented.

It creates a race between API use (e.g. `tc.direction = “inactive”`, `pc.removeTrack(track)`) and (perfect) renegotiation. Implementing it now would break web compat. **Let’s revert it!**

@stefhak was right that:

- `direction` reflects this side's preference in offers and answers
- `currentDirection` reflects the net negotiated direction

E.g. it's normal and informative for them to differ, and **it keeps negotiation from being lossy (answer affects offer)**

- A `direction` of `”sendrecv”` means we have stuff to send and are open to receiving
- A `currentDirection` of `”sendonly”` means the other side has nothing to send ATM

In this view of them as independent attributes, updating them should be deterministic to the app.

It's the nature of negotiation that changes on one side don't always produce a net change in result.

WebRTC-PC Simulcast Issues

- Issue 2722, Issue 2723, Issue 2724 originate from contradictions between RFC 8853 and WebRTC-PC Sections 4.4.1.5 and 5.4.1.
- Section 4.4.1.5 says:

5. If applying *description* leads to modifying a transceiver *transceiver*, and *transceiver*.[[Sender]].[[SendEncodings]] is non-empty, and not equal to the encodings that would result from processing *description*, the process of applying *description* fails. This specification does not allow remotely initiated RID renegotiation.

5. If the description attempted to renegotiate RIDs, as described above, then reject *p* with a newly created InvalidAccessError and abort these steps.

Issue 2722: sRD(offer) completely overwrites pre-existing transceiver.[[Sender]].[[SendEncodings]]

- The language that describes how to handle simulcast in a remote offer says that `[[SendEncodings]]` is completely replaced based on the rids in the simulcast attribute.
  - While this works fine for transceivers that are not yet associated, for already associated transceivers (which have already populated `[[SendEncodings]]`), this is not appropriate.
  - [BA] Over-writing is prohibited in Section 4.4.1.5.
- We need to specify what happens on sRD(offer) when there is already an associated transceiver.
  - Since we (rightly) allow sRD(answer) to remove pre-existing rids, we probably need to allow sRD(offer) to remove pre-existing rids as well (since the base simulcast spec requires the answerer to handle this situation).
  - We also need to ensure that the language around createAnswer does the right thing if the offer tries to add a rid (ie; the answer will not contain that new rid).

Issue 2722: sRD(offer) completely overwrites pre-existing transceiver.[[Sender]].[[SendEncodings]]

- PR 2155 over-writes existing transceiver:

```
+                          <p>If a suitable transceiver was found (<var>transceiver</var>
+                          is set) and <var>sendEncodings</var> is non-empty, set
+                          <var>transceiver</var>.<a>[[\Sender]]</a>.<a>[[\sendEncodings]]</a>
+                          to <var>sendEncodings</var>, and set
+                          <var>transceiver</var>.<a>[[\Sender]]</a>.<a>[[\LastReturnedParameters]]</a>
+                          to <code>null</code>.</p>
```

- Does the recommendation make sense?
- Should we mark this Issue “Ready for PR”?

§ 5.4.1 Simulcast functionality

Simulcast functionality is provided via the `addTransceiver` method of the `RTCPeerConnection` object and the `setParameters` method of the `RTCRtpSender` object.

The `addTransceiver` method establishes the **simulcast envelope** which includes the maximum number of simulcast streams that can be sent, as well as the ordering of the `encodings`. While characteristics of individual simulcast streams can be modified using the `setParameters` method, the simulcast envelope cannot be changed. One of the implications of this model is that the `addTrack()` method cannot provide simulcast functionality since it does not take `sendEncodings` as an argument, and therefore cannot configure an `RTCRtpTransceiver` to send simulcast.

Another implication is that the answerer cannot set the simulcast envelope directly. Upon calling the `setRemoteDescription` method of the `RTCPeerConnection` object, the simulcast envelope is configured on the `RTCRtpTransceiver` to contain the layers described by the specified session description. Once the envelope is determined, layers cannot be removed. They can be marked as inactive by setting the `active` member to `false` effectively disabling the layer.

While `setParameters` cannot modify the simulcast envelope, it is still possible to control the number of streams that are sent and the characteristics of those streams. Using `setParameters`, simulcast streams can be made inactive by setting the `active` member to `false`, or can be reactivated by setting the `active` member to `true`. Using `setParameters`, stream characteristics can be changed by modifying attributes such as `maxBitrate`.

Issue 2723: The prose around "simulcast envelope" falsely implies that simulcast encodings can never be removed (cont'd)

- Spec says "Once the envelope is determined, layers cannot be removed.", but the language for sRD(answer) says that if rids are rejected by an answer, they are removed.

2. If *description* rejects any of the offered layers, then remove the dictionaries that correspond to rejected layers from *transceiver*.`[[Sender]]`.`[[SendEncodings]]`.

[BA] This doesn't appear to be a contradiction to me, since the envelope is set via sRD(), not before.

- There are a couple of ways to fix this:
  1. We remove this assurance from the section on "simulcast envelope", or
  2. We only allow the first answer to remove rids from [[SendEncodings]].

Disallowing an answer to remove rids on a previously negotiated sender is probably not appropriate, since this would violate the simulcast spec, which requires the offerer to handle this case regardless of whether this is the initial negotiation or not. I think option 1 is the correct course of action here.

Issue 2723: The prose around "simulcast envelope" falsely implies that simulcast encodings can never be removed (cont'd)

- What does the WG want to do?
  - Does the WG believe that there is a contradiction in the spec?
  - Is there an interest in enabling re-negotiation?

Issue 2724: The language around setting a description appears to prohibit renegotiation of RIDs (Jan-Ivar)

"This specification does not allow remotely initiated RID renegotiation." was added in #2314.

This spec simultaneously allows answers to reject simulcast layers in an (at least ***initial***) offer.

Together this means running the same O/A again should *succeed* provided the net result is the same the 2nd time. **Agree? Or should new offers face the envelope narrowed by answers and fail?**

RFC8853's example is an offer to send 3 layers, with an answer to receive 2. In WebRTC,

- ✓ Should a subsequent identical O/A succeed because the net result is the same? **Yes/No?**
- ❏ What if the answer rejects 2 layers the second time, resulting in 1 layer? **Yes/No?**
- ❏ What if the answer doesn't reject anything the second time, resulting in 3 layers? **Yes/No?**
- ❏ What if the offer only has 2 layers the second time, does it succeed? **Yes/No?**
- ❏ Does failing an answer that rejects a previously negotiated layer violate RFC8853? **Yes/No?**
- ❏ Does failing an offer that has removed a previously negotiated layer violate RFC8853? **Yes/No?**

RFC 8853 “Using Simulcast in SDP and RTP Sessions”

- Section 4 Overview

```
a=simulcast:send 1;2,3 recv 4
```

- If this line is included in an SDP offer, the "send" part indicates the offerer's capability and proposal to send two simulcast RTP streams.
- Each simulcast stream is described by one or more RTP stream identifiers (rid-ids), and each group of rid-ids for a simulcast stream is separated by a semicolon (";").
- When a simulcast stream has multiple rid-ids that are separated by a comma (","), they describe ***alternative representations for that particular simulcast RTP stream***. Thus, the "send" part shown above is interpreted as an intention to send two simulcast RTP streams. The first simulcast RTP stream is identified and restricted according to rid-id 1.
- The second simulcast RTP stream can be sent as ***two alternatives***, identified and restricted according to rid-ids 2 and 3.
- The "recv" part of the line shown here indicates that the offerer desires to receive a single RTP stream (no simulcast) according to rid-id 4.

RFC 8853 “Using Simulcast in SDP and RTP Sessions”

- Section 5.3.2 Creating the SDP Answer
    - An answerer that receives an offer with simulcast containing an "a=simulcast" attribute listing ***alternative rid-ids*** MAY keep all the alternative rid-ids in the answer, but it MAY also choose to remove any nondesirable ***alternative rid-ids*** in the answer.
    - The answerer MUST NOT add any ***alternative rid-ids*** in the "send" direction in the answer that were not present in the offer receive direction. The answerer MUST be prepared to receive any of the receive-direction rid-id alternatives and MAY send any of the "send"-direction alternatives that are part of the answer.
    - An answerer that receives an offer with simulcast that lists a number of simulcast streams MAY reduce the number of simulcast streams in the answer, but it MUST NOT add simulcast streams.
- Section 5.3.3 Offerer processing the SDP Answer
    - An offerer that receives an answer where some ***rid-id alternatives*** are kept MUST be prepared to receive any of the kept "send"-direction ***rid-id alternatives*** and MAY send any of the kept "receive"-direction rid-id alternatives.
    - An offerer that receives an answer where some of the rid-ids are removed compared to the offer MAY release the corresponding resources (codec, transport, etc) in its "receive" direction and MUST NOT send any RTP packets corresponding to the removed rid-ids.
- RFC 8853 does not prohibit an answer from changing the order of the rids.
- RFC 8853 does not prohibit a re-offer from changing the order of the rids.

Issue 2724: The language around setting a description appears to prohibit renegotiation of RIDs

- Section 4.4.1.5:
  - "5. If the description attempted to renegotiate RIDs, as described above, then reject p with a newly created `InvalidAccessError` and abort these steps."
- This prohibits a local re-offer from adding or removing RIDs.
- However, RFC 8853 indicates that an offerer cannot refuse to honor a remote answer that rejects a previously negotiated RID.
  - RFC 8853 Section 5.3.2:
    - “An answerer that receives an offer with simulcast that lists a number of simulcast streams MAY reduce the number of simulcast streams in the answer, but it MUST NOT add simulcast streams.”
  - RFC 8853 Section 5.3.4:
    - “Offers inside an existing session follow the same rules as for initial SDP offer, with these additions:”

Issue 2724: The language around setting a description appears to prohibit renegotiation of RIDs (cont'd)

- RFC 8853 also indicates that an answerer can't refuse to honor a remote offer because it removed a previously negotiated RID.
  - RFC 8853 Section 5.3.3:
    - "An offerer that receives an answer where some rid-id alternatives are kept MUST be prepared to receive any of the kept "send"-directionrid-id alternatives and MAY send any of the kept "receive"-direction rid-id alternatives.
    - An offerer that receives an answer where some of the rid-ids are removed compared to the offer MAY release the corresponding resources (codec, transport, etc) in its "receive" direction and MUST NOT send any RTP packets corresponding to the removed rid-ids."
  - RFC 8853 Section 5.3.4:
    - "Creation of SDP answers and processing of SDP answers inside an existing session follow the same rules as described above for initial SDP offer/answer."

Issue 2724: The language around setting a description appears to prohibit renegotiation of RIDs (cont'd)

- What does the WG think?

Discussion (**End Time: 09:30 AM**)

-

Yellow-Bellied Slider Turtle

Thank you

Special thanks to:

WG Participants, Editors & Chairs