W3C WebRTC/MediaCapture WG Meeting

April 4, 2017
1 PM PDT

Chairs: Harald Alvestrand
Stefan Hakansson
Bernard Aboba

W3C WG IPR Policy

- This group abides by the W3C patent policy https://www.w3.org/Consortium/Patent-Policy-20040205
- 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 interim meeting of the W3C WebRTC WG!
- During this meeting, we hope to make progress on outstanding issues within webrtc-pc and (if time allows) mediacapture-main and webrtc-stats
- Editor’s Draft updates to follow meeting

webrtc-pc

- As announced, we are targeting to submit a transition (to CR) request before end of April!
- Of course we need to deal with open Issues before - hoping this meeting will be fruitful
- Reminder: we would like feedback from implementers on features that are not on their radar for implementation

About this Virtual Meeting

Information on the meeting:

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/April_4_2017
- Link to latest drafts:
  - https://rawgit.com/w3c/mediacapture-main/master/getusermedia.html
  - https://rawgit.com/w3c/webrtc-pc/master/webrtc.html
  - https://rawgit.com/w3c/webrtc-stats/master/webrtc-stats.html
- Link to Slides has been published on WG wiki
- Scribe? IRC http://irc.w3.org/ Channel: #webrtc
- The meeting is being recorded.
- WebEx info here

For Discussion Today

- **WebRTC-PC**
  - **Pull Requests**
    - Issue 526/PR 1011: NetworkError is not defined and might not be needed (Bernard)
    - Issue 1021/PR 1109: Configurable packetization interval (Taylor)
    - Issue 1085/PR 1098: RTCRtpContributingSource.audioLevel not guaranteed to be in sync with audio playout (Taylor)
    - Issue 1091/PR 1099: When exactly is an SSRC RTCRtpContributingSource object updated? (Taylor)
  - **Issues**
    - Issue 1101: RTCRtpContributingSource naming (Taylor)
    - Issue 1073: Which members in “sendEncodings” are used (Taylor)
    - Issue 763: Handling of Simulcast Errors (Bernard)
    - Issue 1086: Make legacy API optional to implement (Stefan)
    - Issue 1092: DTLS failures (Bernard)

For Discussion Today if time permits

- **Media Capture**
  - **Pull Requests**
    - **Issue 434/PR 440: Disable user media by default in cross-origin iframes (Stefan)**
- **WebRTC Stats**
  - **Pull Requests**
    - **Issue 189 / PR 191: Remove isRemote (jib)**

WebRTC PC Pull Requests

- Issue 526/PR 1011: NetworkError is not defined and might not be needed (Bernard)
- Issue 1021/PR 1109: Configurable packetization interval (Taylor)
- Issue 1085/PR 1098: RTCRtpContributingSource.audioLevel not guaranteed to be in sync with audio playout (Taylor)
- Issue 1091/PR 1099: When exactly is an SSRC RTCRtpContributingSource object updated? (Taylor)

Issue 526/PR 1011: NetworkError is not defined and might not be needed (Bernard)

- Section 6.2 (RTCDataChannel) refers to NetworkError:
  - “If the transport was closed with an error, fire a NetworkError event at channel.”
  - “When the user agent determines that an `RTCDataChannel`'s underlying data transport cannot be created, the user agent must queue a task to run the following steps: … 3. Fire a NetworkError event at channel.”
- Section 13 (Event Summary) for RTCDataChannel:
  - close event (Event interface) fired when “The `RTCDataChannel` object's underlying data transport has been closed.”
  - error event fired when “Any error occured from the data channel.”
- Section 12 defines RTCError
  - Attributes: errorDetail, sdpLineNumber, httpRequestStatusCode, message, name
  - Currently not referenced in Sections 6.2 or 13.

Issue 526/PR 1011: NetworkError is not defined (cont'd)

- Questions:
  - When the RTCDataChannel transport is closed with an error, what event is fired?
  - When an RTCDataChannel error occurs (above the transport), what happens?
    - Example: The RTCDataChannel is closed with an error if there aren't enough stream IDs.
- Comparison to WebSockets:
  - Websockets: Simple error event, CloseEvent has wasClean, code and reason attributes.
  - WebRTC data channel: RTCError event with multiple attributes, simple close event.

Issue 526/PR 1011: NetworkError is not defined (cont'd)

- From https://html.spec.whatwg.org/multipage/comms.html#handler-websocket-onclose :

When *the WebSocket connection is closed*, possibly *cleanly*, the user agent must queue a task to run the following substeps:

1. Change the readyState attribute's value to CLOSED (3).
2. If the user agent was required to *fail the WebSocket connection*, or if the *the WebSocket connection was closed* after being flagged as full, fire an event named error at the WebSocket object. [WSP]
3. Fire an event named close at the WebSocket object, using CloseEvent, with the wasClean attribute initialized to true if the connection closed *cleanly* and false otherwise, the code attribute initialized to *the WebSocket connection close code*, and the reason attribute initialized to the result of applying UTF-8 decode without BOM to *the WebSocket connection close reason*. [WSP]

Issue 526/PR 1011: NetworkError is not defined (cont'd)

- Proposal:
  - When the RTCDataChannel transport is closed with an error, fire the RTCError event with errorDetail = "sctp-failure", sctpCauseCode = SCTP Cause Code, and message set to the SCTP Cause-Specific-Information, possibly with additional text.
  - When an RTCDataChannel error occurs above the transport, fire the RTCError event with errorDetail = "datachannel-failure".
  - Do we want additional CloseEvent attributes as in Websockets?

Issue 526/PR 1011: NetworkError is not defined (cont'd)

- SCTP Operation Error (ERROR) Chunk provides one or more Cause Code/Cause-Specific-Information blocks. From IANA registration page:

Error Cause Codes

Registration Procedure(s)
Specification Required
Reference
[RFC4960]
Available Formats
CSV

Value	Cause Code	Reference
1	Invalid Stream Identifier	[RFC4960]
2	Missing Mandatory Parameter	[RFC4960]
3	Stale Cookie Error	[RFC4960]
4	Out of Resource	[RFC4960]
5	Unresolvable Address	[RFC4960]
6	Unrecognized Chunk Type	[RFC4960]
7	Invalid Mandatory Parameter	[RFC4960]
8	Unrecognized Parameters	[RFC4960]
9	No User Data	[RFC4960]
10	Cookie Received While Shutting Down	[RFC4960]
11	Restart of an Association with New Addresses	[RFC4960]
12	User Initiated Abort	[RFC4460]
13	Protocol Violation	[RFC4460]
14-159	Unassigned	
160	Request to Delete Last Remaining IP Address	[RFC5061]
161	Operation Refused Due to Resource Shortage	[RFC5061]
162	Request to Delete Source IP Address	[RFC5061]
163	Association Aborted due to illegal ASCONF-ACK	[RFC5061]
164	Request refused - no authorization	[RFC5061]
165-260	Unassigned	
261	Unsupported HMAC Identifier	[RFC4895]
262-65535	Unassigned	

Issue 1021/PR 1109: Configurable packetization interval (Taylor)

- Decision at previous interim: ptime should be configurable, but only as a hint. No need to list all supported ptimes in capabilities.
- Content of PR:

`ptime` of type unsigned long

For an `RTCRtpSender`, indicates the preferred duration of media represented by a packet in milliseconds for this encoding. The user agent must use this duration if possible, and otherwise use the closest available duration. This value must take precedence over any "ptime" attribute in the remote description, whose processing is described in [JSEP]. Note that the user agent must still respect the limit imposed by any "maxptime" attribute, as defined in [RFC4566], Section 6.

Issue 1085/PR 1098: RTCRtpContributingSource.audioLevel not guaranteed to be in sync with audio playout (Taylor)

- RTCRtpContributingSources are currently updated whenever an RTP packet is received. If they're being used to drive a UI that shows speaker audio levels, that UI isn't guaranteed to be in sync with the audio being played out.
- **Proposal:** Change "updated when packet is received" to "updated at playout time," for some definition of "playout time."
- PR 1098 is an initial attempt at this. Excerpt: "When the first audio frame contained in an RTP packet is delivered to the RTCRtpReceiver's MediaStreamTrack for playout, the relevant RTCRtpContributingSource objects are updated."

Issue 1091/PR 1099: When exactly is an SSRC RTCRtpContributingSource object updated? (Taylor)

- RTCRtpContributingSource objects are returned for both CSRCs and SSRCs; in what circumstances are the SSRC objects updated?
- Specification currently states:
  - “**If the RTP packet contains no CSRCs**, then the RTCRtpContributingSource object corresponding to the SSRC is updated”
- However, sometimes it may be useful to have a per-SSRC audio level, even if a packet contained CSRCs.
- **Proposal**: Always update the SSRC object. This shouldn’t add any complexity to the specification or implementations, and may be useful to applications.

WebRTC PC Issues

- Issue 1101: RTCRtpContributingSource naming (Taylor)
- Issue 1073: Which members in “sendEncodings” are used (Taylor)
- Issue 763: Handling of Simulcast Errors (Bernard)
- Issue 1086: Make legacy API optional to implement (Stefan)
- Issue 1092: DTLS failures (Bernard)

Issue 1101: RTCRtpContributingSource naming (Taylor)

- getContributingSources returns objects representing both CSRCs and SSRCs. SSRCs, by definition, are not contributing sources, so this naming is confusing.
- Should we:
  - Call the method “getSources” instead. It returns both contributing (CSRC) and synchronization (SSRC) sources; makes sense.
  - Add a separate “getSynchronizationSources” method.
  - Leave things the way they are and document the discrepancy.

Issue 1073: Which members in “sendEncodings” are used (Taylor)

- “sendEncodings” is currently only specified to do one thing: affect the number of “a=rid” lines, and the RID values used.
- However, this conflicts with the simulcast example, which sets both RID values and encoding parameters:

```
pc.addTransceiver(track, {
  direction: "sendrecv"
  encodings: [
    { rid: "f" },
    { rid: "h", scaleDownResolutionBy: 2.0},
    { rid: "q", scaleDownResolutionBy: 4.0}
  ]
});
```

Issue 1073: Which members in “sendEncodings” are used (cont’d)

- So, should “sendEncodings” just be used for RIDs?
  - The simulcast example would then need to a do a full “setLocalDescription”, “setRemoteDescription”, “sender.getParameters”, “sender.setParameters”.
- Or should it function as an early “RTCRtpSender.setParameters”?
  - If so, what members of RTCRtpEncodingParameters are used?

Issue 1073: Which members in “sendEncodings” are used (cont’d)

- Reminder of what’s contained in RTCRtpEncodingParameters:

```
dictionary RTCRtpEncodingParameters {
    unsigned long       ssrc;
    RTCRtpRtxParameters rtx;
    RTCRtpFecParameters fec;
    RTCDtxStatus        dtx;
    boolean             active;
    RTCPriorityType     priority;
    unsigned long       maxBitrate;
    unsigned long       maxFramerate;
    DOMString           rid;
    double              scaleResolutionDownBy = 1;
};
```

Issue 763: Handling of Simulcast Errors (Bernard)

- How does an application discover how many simulcast streams can be sent?
  - If “too many” simulcast streams are requested, pc.addTransceiver(init) or can throw an InvalidParameters exception or promise rejected in transceiver.sender.setParameters().
  - In existing implementations, simulcast is supported for some codecs, but not others.
- What might we want to know?
  - For a given codec, maximum number of simulcast streams that can be sent.
    - Makes it possible to indicate that simulcast is unsupported for a codec.
  - Are there implementations with a limit to the aggregate simulcast streams that can be sent?
- Does it make sense to add a maxSimulcastStreams attribute in RTCRtpCodecCapability?

```
partial dictionary RTCRtpCodecCapability {
    Unsigned long maxSimulcastStreams = 0;
};
```

Issue 1086: Make Legacy API Optional to Implement (Stefan)

- Proposed by youennf
  - Not enthusiastic about carrying the legacy APIs in WebKit
- However, Chrome counters say the legacy API is used a lot
- Is there any precedence on what to do? (looking at Dom :))
  - Foolip: “Most specs have some bits of API that aren't yet implemented everywhere without that being considered a spec violation, so in that way implementations are already free to prioritize implementing other things over the callback-based APIs.”

Issue 1092: DTLS Failures (Bernard)

- RTCDtlsTransport object currently has `onstatechange` EventHandler, and RTCDtlsTransportState of “failed”.
- No onerror EventHandler.
- Question: is there any actionable information that can be provided to applications without compromising security? Example:
  - Mismatched cryptosuites (e.g. browser offers only ECDSA certificates, peer is a mobile application that only supports DTLS 1.0 with RSA certificates).

Issue 1092: DTLS Failures (cont'd)

From Websockets specification:

User agents must not convey any failure information to scripts in a way that would allow a script to distinguish the following situations:

- *A server whose host name could not be resolved.*
- *A server to which packets could not successfully be routed.*
- *A server that refused the connection on the specified port.*
- *A server that failed to correctly perform a TLS handshake (e.g., the server certificate can't be verified).*
- *A server that did not complete the opening handshake (e.g. because it was not a WebSocket server).*
- *A WebSocket server that sent a correct opening handshake, but that specified options that caused the client to drop the connection (e.g. the server specified a subprotocol that the client did not offer).*
- *A WebSocket server that abruptly closed the connection after successfully completing the opening handshake.*

Issue 1092: DTLS Failures (cont'd)

In all of these cases, the the WebSocket connection close code *would be 1006, as required by the WebSocket Protocol specification. [WSP]*

Allowing a script to distinguish these cases would allow a script to probe the user's local network in preparation for an attack.

In particular, this means the code 1015 is not used by the user agent (unless the server erroneously uses it in its close frame, of course).

The task source for all tasks queued in this section is the WebSocket task source.

Media Capture Pull Requests

- **Issue 434/PR 440: Disable user media by default in cross-origin iframes (Stefan)**

Issue 434/PR 440: Disable user media by default in cross-origin iframes (Stefan)

- In current spec, iframes only get to use getUserMedia if the parent allows it to (and the parent itself is allowed to)
- The ‘allowusermedia’ attribute is used, e.g.
  - <iframe allowusermedia …..
- Proposal to instead individually allow camera, microphone and speakers(!) - won’t talk about speakers here, it’s another spec
  - <iframe allow=”camera microphone” ….
  - Spec to refer to: https://wicg.github.io/feature-policy/
- Both methods seem to be moving a bit, and which will prevail is not crystal clear
- PR to introduce proposed method at https://github.com/w3c/mediacapture-main/pull/440 (note: to be updated)

WebRTC Stats Pull Requests

- Issue 189 / PR 191: Remove isRemote (jib)

Stats Issue 189/ PR 191 Remove isRemote (jib)

- **Today:** Can't enumerate RTCRtpStreamStats without checking `isRemote`.
- Remote/local dictionaries are asymmetric. Must read prose or check to learn what's where.
- **Instead:** New types: `"remote-inbound-rtp"`, `"remote-outbound-rtp"`. New dictionaries:

```
dictionary RTCInboundRTPStreamStats : RTCReceivedRTPStreamStats { // Common base bikeshed name
    DOMString           remoteId; // looks up RTCRemoteOutboundRTPStreamStats
    unsigned long       framesDecoded;
};
dictionary RTCRemoteInboundRTPStreamStats : RTCReceivedRTPStreamStats {
    DOMString           localId;  // looks up RTCOutboundRTPStreamStats
    double              roundTripTime;
};
dictionary RTCOutboundRTPStreamStats : RTCSentRTPStreamStats {
    DOMString           remoteId; // looks up RTCRemoteInboundRTPStreamStats
    double              targetBitrate;
    Unsigned long       framesEncoded;
    double              totalEncodeTime;
};
dictionary RTCRemoteOutboundRTPStreamStats : RTCSentRTPStreamStats {
    DOMString           localId;  // looks up RTCInboundRTPStreamStats
    DOMHighResTimeStamp remoteTimestamp;
};
```

Thank you

Special thanks to:

Harald!!!

W3C/MIT for WebEx

WG Participants, Editors & Chairs