# W3C WebRTC WG Meeting

June 4, 2020
9:30 AM Pacific Time

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 June interim meeting of the W3C WebRTC WG!
  - During this meeting, we will talk about new work and make progress on privacy and security concerns.

# About this Virtual Meeting

## Information on the meeting:

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/June_4_2020
- Link to latest drafts:
  - https://w3c.github.io/mediacapture-main/
  - 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-stats/
  - https://www.w3.org/TR/mst-content-hint/
  - https://w3c.github.io/webrtc-nv-use-cases/
  - https://w3c.github.io/webrtc-dscp-exp/
  - 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 being recorded.

# Issues for Discussion Today

- Simulcast testing update (Bernard)
- Insertable Streams (Harald)
- WebRTC-PC
  - Issue 2534: TCP portscanning
- Media Capture & Streams
  - Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (Jan-Ivar)
- Media Capture Output
  - Issue 86: API to request audio output device selection (Youenn)
  - Issue 87: Setting the audio output for a whole page (Youenn)
- Media Capture Automation (Youenn)

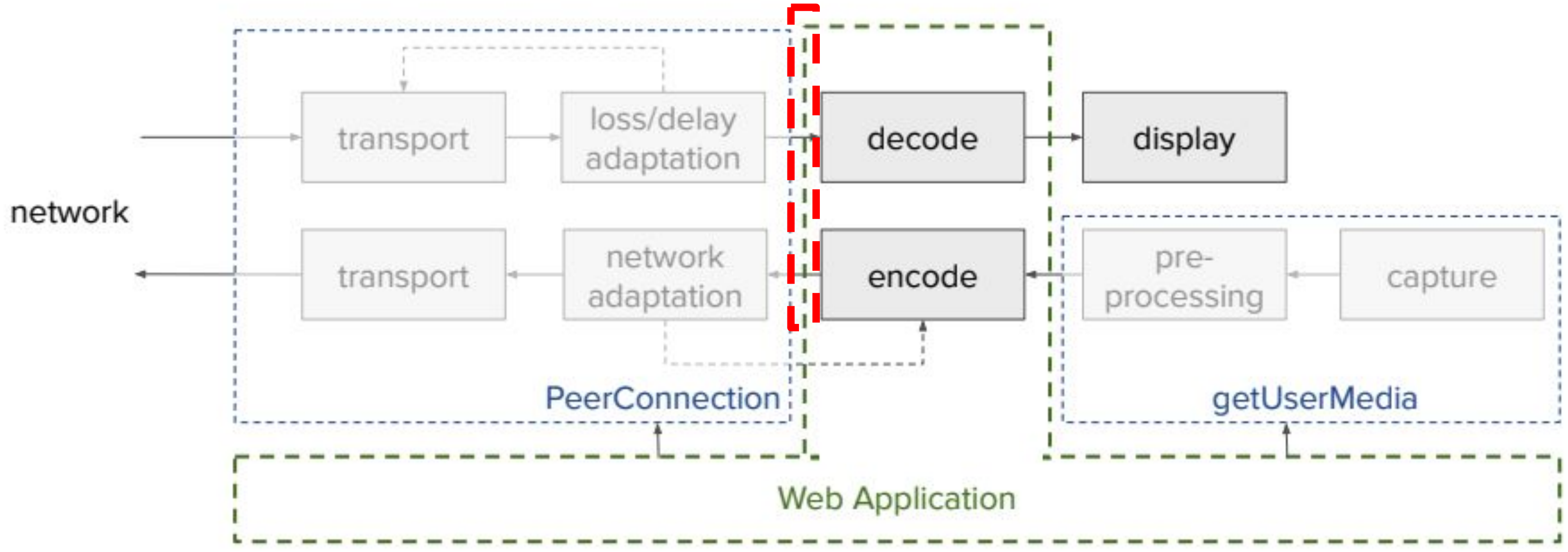
## Simulcast Testing Update (Bernard)

- Based on “simulcast playground” and a proof-of-concept from Florent, Fippo has contributed some simulcast tests:
  - `basic.https.html`: loopback test demonstrating the ability to send simulcast and to receive and render each encoding (VP8+H264 variants tested)
    - Tests addTransceiver support for multiple encodings.
    - Tests ability to send RID and receive MID header extension.
  - `getStats.https.html`: demonstrates that stats corresponding to the rids are retrievable.
- Section 5.4.1 now annotated to indicate additional test coverage.

# Simulcast Testing (cont'd)

- How can we extend WPT coverage?
  - Codec-specific tests:  test that simulcast works for each video codec (VP8, VP9, H.264/AVC, AV1, etc.)
    - CL landed (simulcast/h264.https.html, vp8.https.html)
  - Tests of RTCRtpEncodingParameters dictionary members:
    - active: test that setting active=false stops sending an encoding, active=true enables sending again. CL landed
    - maxBitrate: test that setting a value results in a bitrate less than the requested value.
    - maxFramerate: test that setting a value results in a framerate less than the requested value. CL under review.
    - scaleResolutionDownBy: test that setting a value results in a resolution scaling approximating the requested value.
- What can't we test in WPT?
  - Robustness: no loss in loopback, so can't really test RRID, RTX, RED or FEC
  - Interop with SFUs or MANEs (requires KITE)
- Comments or suggestions?

# Insertable Streams - Current Idea



# Insertable Streams - Status

Implementation status

- Implemented in Chrome
- Available in Chrome 83 (turn on --enable-experimental-web-platform-features)
- Available as an “origin trial” to any website that registers
- Used for Duo and Jitsi end-to-end additional frame encryption
- Experimented with for a number of other purposes

Specification status

- Explainer and spec document conform ~ to Chrome 84 code
- Metadata information needs are still being teased out
- We would like to have this as part of the Web’s API

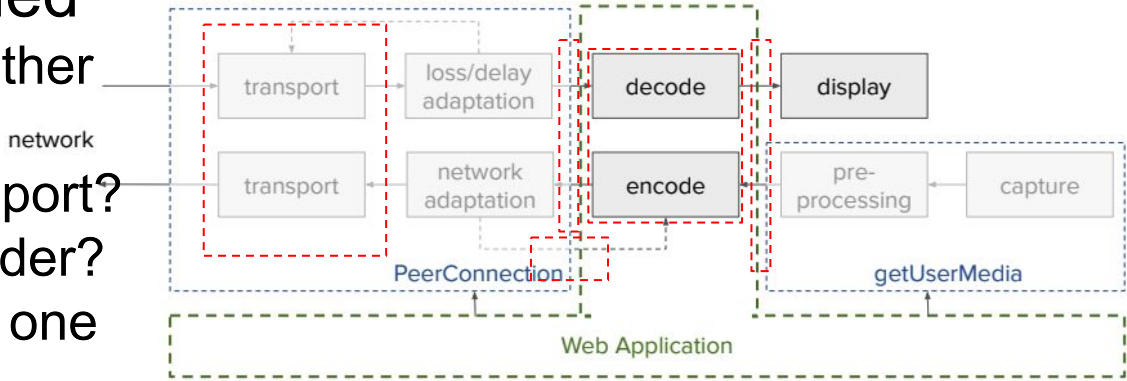
Is this an acceptable starting point?
CfC feedback on the mailing list has been positive.

## Insertable streams - Limitations

- No creating frames (dropping is OK)
  - Implies no input from other than receiver
- No manipulating metadata
- No interactions with feedback signals
  - With hefty data additions, congestion feedback is a concern

# Insertable Streams - Future

- We think this API is worth living with
- Extensions envisioned
  - Feedback path - another stream?
  - Bring your own transport?
  - Bring your own encoder?
- Will keep supporting this one



## Issue 2534: TCP portscanning

Connecting to random TCP ports from your browser and reporting the result back to the page is a Bad Idea for security.

- Portscanning is the most obvious application

Browsers need to be able to say “no”

- Allow policy and define behavior to match

Decide: Error return or “silently don’t connect”?

## Media Capture Automation (Youenn)

- Automated testing is hard...
  - Dealing with user prompts
  - Using real devices is sometimes a problem
- … but is beneficial
  - For browser developers / WPT to improve interoperability
  - For web site authors to ensure they run fine on browsers
- What to test
  - getUserMedia resolve/reject, autoplay policies
  - constraints matching: getUserMedia + applyConstraints
  - devicechange event
  - enumerateDevices filtering

## Media Capture Automation (Youenn)

- Proposal: a dedicated WebDriver API
  - https://youennf.github.io/media-capture-automation/media-capture-automation.html
  - Control capture prompt result (grant/deny)
  - Add/remove fake devices
- Some limited control of fake device capabilities
  - min/max resolution, frame rate, sample rate
- Currently out-of-scope
  - Specifying the content generated by fake devices
    - Use WebAudio/CanvasCapture instead
  - Access to real devices

## Issues for Discussion Today

- Media Capture & Streams
  - Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (Jan-Ivar)
- Media Capture Output
  - Issue 86: API to request audio output device selection (Youenn)
  - Issue 87: Setting the audio output for a whole page (Youenn)

## Issue 669: "user-chooses": Do required constraints make any sense now? (Henrik)

In-chrome pickers competes with in-content pickers. *Where are we headed?*

Today, **“required”** constraints remove devices from the selection.

- Does this make any sense with **“user-chooses”?**
  Example: SD camera faces me, HD camera faces my room. Application prefers HD, but that’s not what the user wants! Why not let the user pick?

To what extent should filtering out devices be allowed in **“user-chooses”**?
More importantly, to what extent do we want to expose deviceIds and labels?

Can of worms?

- deviceIds are used for in-content selection.
- deviceIds are used to avoid prompt when re-visiting website.
- deviceIds are used to toggle cameras (e.g. “front” and “back”).

## Issue 669: "user-chooses": Do required constraints make any sense now? (Henrik)

**Proposal: When using "user-chooses"...**

- deviceId can still be **required** and filter out devices.
- Any other **required** constraints are *ignored if they would reduce the set of devices*.
  - E.g. you may force HD on a HD/LD device, but you may not exclude LD-only devices.

**Flavor A:** Full in-content picker.

- deviceId and labels of *ALL* devices are exposed when permission is granted to *ANY* device.

**Flavor B:** Partial in-content picker.

- Expose current and *minimally sensitive deviceIds* like for "front" and "back" camera.
- For other devices, a special deviceId for "other devices" to cause re-prompt.

**Flavor C:** In-content picking is not supported.

- Deprecate deviceIds. Only user selects.
- Avoid re-prompt on revisit with **{previousDevice:true}** which prefers "whatever I used last time on this domain", may avoid re-prompt.

## Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (jib)

Chrome/Edge & Safari have `info.getCapabilities()` w/info on all devices after gUM.

**Reason:** Lets site enforce its constraints while building picker, or choosing other device outright. Most sites enforce some constraints. **But:** It's a trove of fingerprinting info!

`"user-chooses"` provides feature-parity, without the information leak:

```
await navigator.mediaDevices.getUserMedia({video: constraints, semantics: "user-chooses")
```

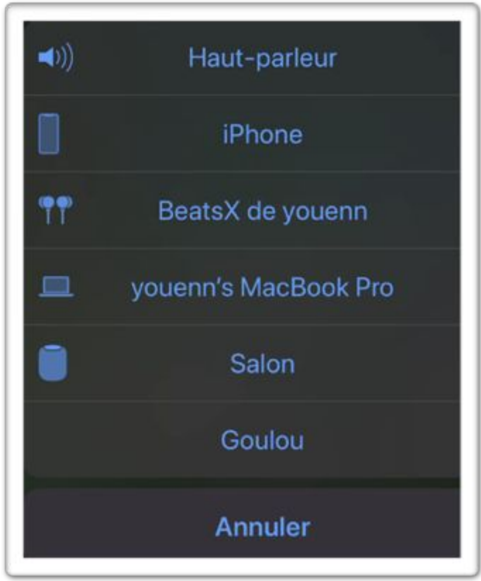
So once #667 merges, can we deprecate `info.getCapabilities()`?

## Issue 86: API to request audio output device selection (Youenn)

- *HTMLMediaElement.setSinkID*
  - But page needs to get device IDs
    - Chrome gives IDs if camera/microphone access is granted
  - Not all pages want camera/microphone
    - Webcast scenario

## Issue 86: API to request audio output device selection (Youenn)

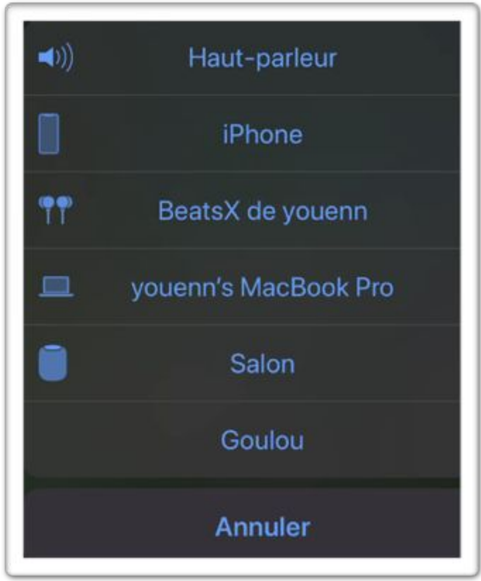
- Proposal: prompt the user for selection
- First option: a new API
    - *partial interface MediaDevices {*
      *Promise<DOMString> selectAudioOutput()*
      *}*
- Selected device gets exposed in enumerateDevices list
- Web page calls *setSinkID*



Chrome UI

## Issue 86: API to request audio output device selection (Youenn)

- Proposal: prompt the user for selection
- Second option: update *HTMLMediaElement.setSinkId*
  - *setSinkId* currently rejects if id is not listed in enumerateDevices list
  - Change spec to prompt the user if no id provided or id is not listed



Chrome UI

## Issue 86: API to request audio output device selection (Youenn)

- Device selection without prompts
  - Use already discovered devices
  - Use statically defined device IDs
    - 'earpiece', 'speaker', 'default', 'none'
  - Select headset whose microphone is in use
    - Expose audio output device if its related microphone is already exposed or used for capture

## Issue 87: Setting the audio output for a whole page (Youenn)

- *HTMLMediaElement.setSinkID*
  - But requires calling *setSinkID* on all media elements
    - Plus no WebAudio APIs yet
    - Plus no third-party iframe control
  - Most applications want a single audio output

## Issue 87: Setting the audio output for a whole page (Youenn)

- Proposal: Add an API to set audio output for the whole page

```
partial interface MediaDevices {
    Promise<> setSinkId(DOMString)
}
```

  - *HTMLMediaElement.setSinkId* to override the page default
- Why?
  - Simpler, less error prone and good enough for most apps
  - Easier to present to users at OS level
  - Easier to implement

## For extra credit

**Sergio**

## Thank you

Special thanks to:

WG Participants, Editors & Chairs

The bird