Simulcast in WebRTC 1.0

at the 2015 f2f

Understanding Simulcast

Scenario: browser talking to a Selective Forwarding Unit (SFU). Typically not needed for P2P scenarios.

Sender: Sends multiple encodings of the same source, differing in framerate, resolution or both. Streams can use distinct PT and SSRC, or same PT and distinct SSRC.

SFU: selects which sender stream to forward to a receiver.

Receiver: If PT/SSRC does not change when switching between sender streams, not "simulcast" from receiver point of view. This is the behavior we assume for this discussion.

Choices for Discussion

```
Option A: Tell createOffer to produce simulcast SDP

Option B: Call RtpSender.setParameters()

Option C: Clone tracks (what we have today)
```

Option A: One Tiny Knob + MMUSIC-defined SDP

```
partial interface RTCRtpSender {
  attribute unsigned short maxSimulcastCount;
};
```

`maxSimulcastCount` of type unsigned short

> This attribute controls the number of simulcast streams that will be offered for the specific RTCRtpSender. The actual number of streams used for this sender will depend on the answer that is passed to setRemoteDescription

What does this knob do?

The only effect of this knob is to add an “a=simulcast” line to the corresponding m-section, and add on some additional PTs* in the offer, so that the answerer has room to specify the parameters associated with additional simulcast encodings.

All other simulcast behavior is controlled by the MCU, following the negotiation described in `draft-ietf-mmusic-sdp-simulcast`.

** Or whatever the mmusic draft settles on*

So what does the offer look like, then?

Based on the current work in MMUSIC, something like:

```
m=video 49300 RTP/AVP 97 98
a=rtpmap:97 H264/90000
a=rtpmap:98 H264/90000
a=fmtp:97 profile-level-id=42c01f; max-fs=3600; max-mbps=108000
a=fmtp:98 profile-level-id=42c01f; max-fs=3600; max-mbps=108000
a=imageattr:97 send [x=[128:16:1280],y=[72:9:720]] recv
    [x=[128:16:1280],y=[72:9:720]]
a=imageattr:98 send [x=[128:16:1280],y=[72:9:720]]
a=simulcast send 97;98 recv 97
```

Harald's Further Simplification

"Why do you even need a knob?"

This could well work without any API surface: all video m-lines could simply include room for a preset number of simulcast encodings (Two? Three?)

This means that WebRTC implementations simply use the SDP negotiation described in `draft-ietf-mmusic-sdp-simulcast` to activate simulcast, with no additional API needed.

Option A Example

```
var sender = pc.addTrack(...);
sender.maxSimulcastCount = 3;
var offer = pc.createOffer();  // Offer now has 3x the PTs
// ... normal setLocalDescription/setRemoteDescription
var count = sender.getParameters().encodings.length; // count == 3
```

Option A Pros/Cons

Pros

- Simple for JS, just one field
- Simulcast streams are explicitly identified so capture/encoding adjustments can be coordinated

Cons

- Relies on simulcast SDP, which isn't done
    - *Opinions differ on how far it is from done, however*
- Doesn't allow per-encoding control (without SDP munging)

Option B: Call RtpSender.setParameters

```
var sender = pc.addTrack(...);
// ... normal createOffer/setLocalDescription/setRemoteDescription
sender.setParameters({
  encodings: [
    sender.getParameters().encodings[0],
    {resolutionScale: 2},
    {resolutionScale: 4}
  ]
});
```

Option B Pros/Cons

Pros

- No new API points
- Per-encoding control
- No reliance on SDP
- Simulcast streams are explicitly identified so capture/encoding adjustments can be coordinated.
- Streams can use distinct PT/SSRC or just distinct SSRCs

Cons

Option C: Clone tracks

```
var track2 = track.clone();
track2.applyConstraints(...);
var track3 = track1.clone();
track3.applyConstraints(...);
pc.addTrack(track);
pc.addTrack(track2);
pc.addTrack(track3);
```

Option C Pros/Cons

Pros

- No new API points. In fact, no spec or implementation changes at all!
- Per-encoding control
- Some applications are already doing it this way

Cons

- More advanced JS
- Existing implementations do not coordinate capture/encoding adjustments between streams.

Path Forward?

What do we do with this?

A. Simple knob + MMUSIC SDP

B. RtpSender.setParameters(encodings * N)

C. Track cloning (AKA do nothing)