

DataChannel issues

Randell Jesup
Salvatore Loreto

IETF 85 Atlanta

Congestion Control

- Low latency preferred
 - Fights with media streams
 - Large transfers could kill interactive media
 - Primary requirement

Options

- Default
 - Loss-based, can kill media
 - Can impose a cap on data (% or max)
 - Can allow a minimum bandwidth
 - Can be dynamic and under control of the application, which can get feedback about the current link bandwidth available via the JS API interfaces
- Act as slave to media congestion control, if it exists (% of what media thinks is available)

More Options

- LEDBAT
 - Scavenger protocol
 - Can impose a fixed delay (spec 100ms)
 - Could be tuned to a much lower version, if the algorithm is stable, like 10 or 20ms
 - Downside is that an application can't guarantee any reasonable amount of bandwidth to the data channel; it's unclear if the congestion algorithm for media and LEDBAT would collapse to one getting all or the other getting all
- Future: RMCAT

Initial creation

- Currently we can create channels once the connection is up
- It's proposed that we pre-define channels at connection time (via SDP)
 - Cuts RTT in setup at call-creation time
 - Many applications will have a fixed list of datachannels
- SDP would be in the MMUSIC SCTP/DTLS draft

What is in a particular DataChannel?

- When we have two different apps in a call, they may want to still use datachannels to exchange data in a well-known form – not just application specific
- Label is insufficient, though people would make it work
- Suggestion: add the 'protocol' field from websockets (in addition to label), which is a registered value.

Example

- Channel =
`pc.createDataChannel("label", "protocol",
{ options });`
- Channel =
`pc.createDataChannel("Chat", "application/blah-
chat-proto", {});`
- `pc.onDataChannel(function(channel) {
 if (channel.protocol == "application/blah-chat-
proto") { }
}`

Questions/Discussion