FEC in WebRTC

ietf-rtcweb-fec-01

Changes since -00

- Adopted as WG draft
- Removed section on FEC for Opus in CELT mode
    - By default, Opus will fall back to SILK mode when FEC is needed
- Changed RFC2198 usage with G.711 to MAY
- Added brief discussion of FEC for data channels
    - Since application controls what is sent, it can do its own FEC at the application level

Clarifications

- When sending FEC on its own stream, it is SSRC-multiplexed [RFC5956] with the primary stream, like RTX [Unified Plan]
- At present, there is no defined way to send a FEC stream that protects multiple primary streams
  - Left for future study

Example SDP

```
m=audio 20000 UDP/TLS/RTP/SAVPF 96
a=mid:a1
a=rtpmap:96 opus/48000/2
a=fmtp:96 useinbandfec=1
a=ssrc:1111

m=video 30000 UDP/TLS/RTP/SAVPF 100 110
c=IN IP4 233.252.0.1
a=mid:v1
a=rtpmap:100 VP8/90000
a=rtpmap:110 interleaved-parityfec/90000
a=fmtp:110 L:5; D:10; ToP:0; repair-window:200000
a=ssrc:1234
a=ssrc:2345
a=ssrc-group:FEC-FR 1234 2345
```

Open Issues (1)

- FEC for audio codecs without internal FEC
  - Current recommendation: RFC 2198 redundancy
    - Bitrate increase: 1x
    - Delay increase: 1 packet
  - Question: should we support flexfec/ulpfec?
    - Bitrate increase: 1/N
    - Delay increase: N packets

Open Issues (2)

- Security considerations
  - App can set FEC parameters to cause significant blowup
  - Congestion control should handle this, when used (e.g. rmcat mechanisms or circuit breakers)
  - If congestion control is not used (e.g. legacy audio interop), implementation should impose limits on maximum rate that can be sent
    - e.g. should redundant PCMU stream be allowed?