Extension Encryption

(Preview)

Justin Uberti, Google
Sergio Garcia Murillo, Cosmo
IETF 108

Problem Statement

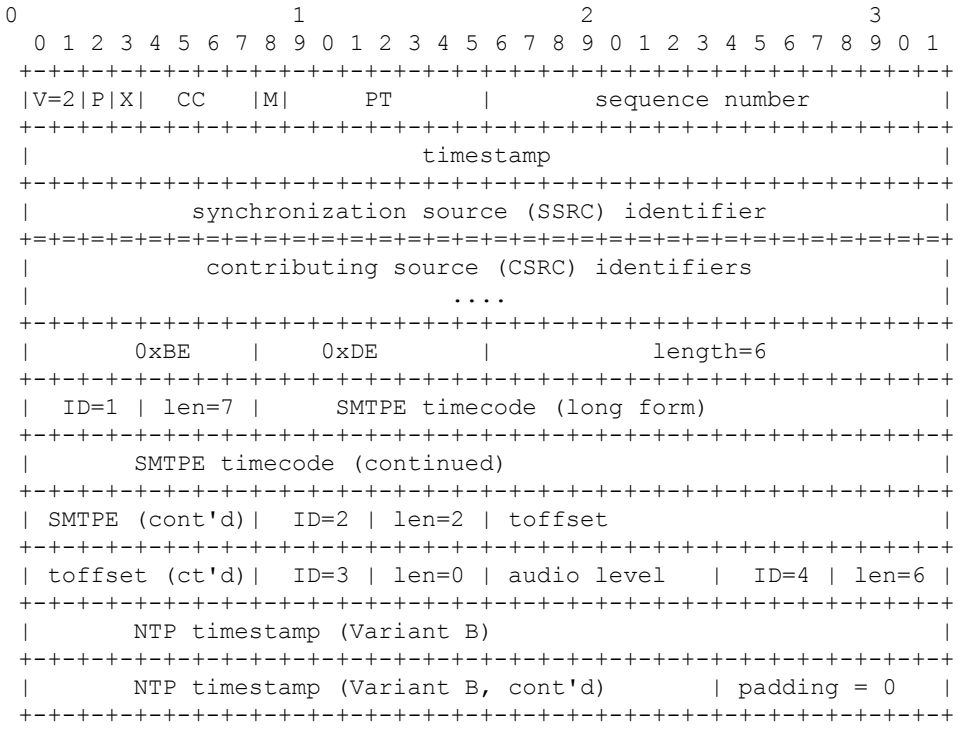
- WebRTC implementations use a lot of extensions (e.g., Safari)

```
a=extmap:1 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=extmap:14 urn:ietf:params:rtp-hdrext:toffset
a=extmap:2 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
a=extmap:13 urn:3gpp:video-orientation
a=extmap:3 http://www.ietf.org/id/draft-holmer-rmcat-transport-wide-cc-extensions-01
a=extmap:12 http://www.webrtc.org/experiments/rtp-hdrext/playout-delay
a=extmap:11 http://www.webrtc.org/experiments/rtp-hdrext/video-content-type
a=extmap:7 http://www.webrtc.org/experiments/rtp-hdrext/video-timing
a=extmap:8 http://tools.ietf.org/html/draft-ietf-avtext-framemarking-07
a=extmap:9 http://www.webrtc.org/experiments/rtp-hdrext/color-space
a=extmap:4 urn:ietf:params:rtp-hdrext:sdes:mid
a=extmap:5 urn:ietf:params:rtp-hdrext:sdes:rtp-stream-id
a=extmap:6 urn:ietf:params:rtp-hdrext:sdes:repaired-rtp-stream-id
```

- Some of these are at least somewhat sensitive
  - e.g., `ssrc-audio-level, video-content-type`
- All of these leak some amount of metadata
  - e.g., application type, HDR support, HW/SW encoder

Current

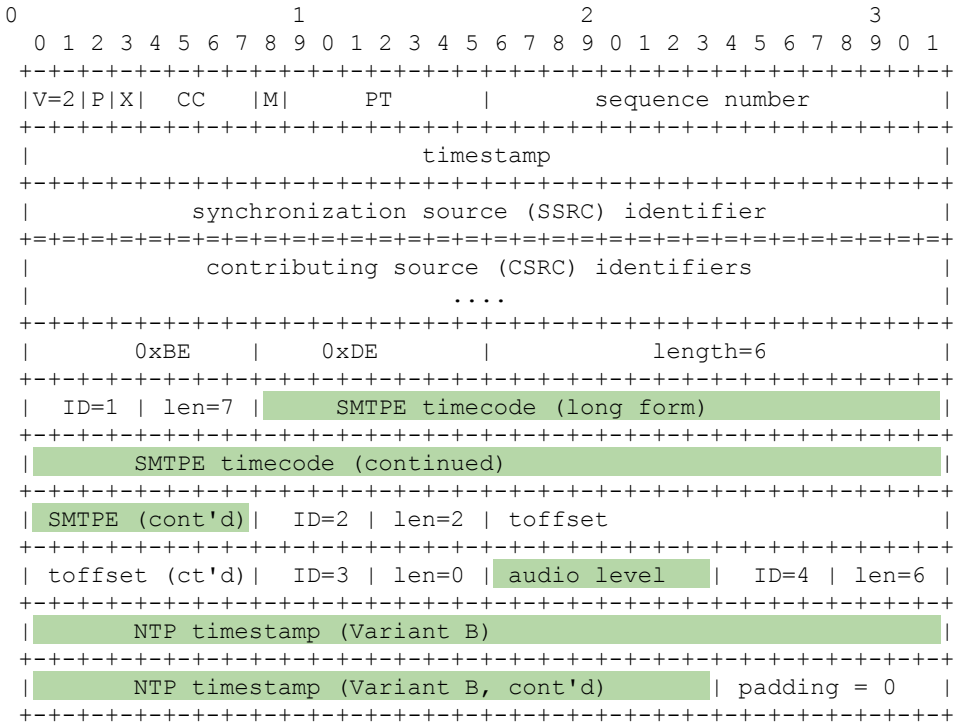
None of the RTP header is encrypted, including extensions



Solution

- To prevent metadata leaks, we should encrypt RTP header extensions, and make this MTI for WebRTC implementations

Option A: RFC 6904



Challenges

- RFC 6904 defines a mechanism to encrypt header extension values, but is complicated; this complexity has hindered adoption
- RFC 6904 also has some technical deficiencies

Opportunity

- Could we find an approach that is both easier to deploy and more secure?