Extension Formatting for the Opus Codec
draft-valin-opus-extension-01

Presenter: Jean-Marc Valin
jmvalin@amazon.com

IETF 117
Extending Opus

• Goals
  - Support all goals of the mlcodec WG
  - Maintain full compatibility with original specification
  - Make it possible to signal extensions in-band
  - Reasonable future-proofing vs efficiency trade-off

• Proposed solution: transmit extensions as Opus padding
  - No change to the meaning of the other bits
  - Extensions will be ignored by older decoder
Why Compatibility Is Preserved

● From RFC6716
  - “The additional padding bytes [...] MUST be set to zero”
  - “The decoder MUST accept any value for the padding bytes”

● All-zero padding is still interpreted as regular padding
Proposed Format

- Concatenation of extensions
- Each extension contains (byte-aligned)
  - 8-bit header (7-bit ID, L flag)
  - Optional length byte(s) (depending on L)
  - Payload
Extension IDs and L Flag

- **ID=0**: Padding
  - $L=0 \rightarrow$ rest of padding must be zeros (original meaning)
  - $L=1 \rightarrow$ no length or payload follows
- **ID=1..31**
  - $L=0 \rightarrow$ no length or payload
  - $L=1 \rightarrow$ one byte of payload follows
- **ID=32..127**
  - $L=0 \rightarrow$ payload fills the rest of the padding
  - $L=1 \rightarrow$ signal payload length
Assigned IDs

- ID=0: Padding
- ID=1: Frame separator
  - Separates frame extensions in multi-frame packets
- ID=2..119: To be assigned with “standards action” policy
- ID=120..127: “Experimental use” policy
SDP

• General parameters:
  - `extensions=<list of IDs>`
  - `sprop-extensions=<list of IDs>`

• Extension-specific parameters:
  - `extN-*=<param>` and `sprop-extN=<param>`
Open Questions

- Allow repeat ID for same frame in a packet?
- Should ordering matter?
- Should we pre-define “unsafe” extensions?
  - Similar to draft-ietf-tsvwg-udp-options