SFrame
E2EE for Video Conferencing

IETF 108 Dispatch

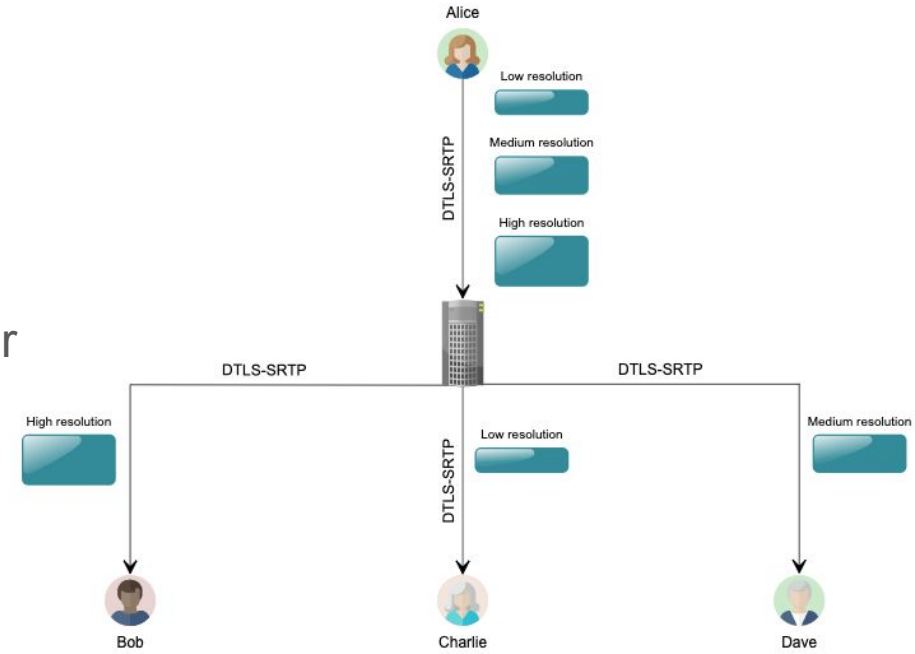
emadomara@google.com

Goals

- Security
  - Secure end to end communications between the end points
- Simplicity
  - Minimize the changes needed in the group media server and end points
- Efficiency
  - Minimized the encryption overhead between the endpoints
- Compatibility
  - Works with existing RTC protocols like WebRTC
  - Works with RTC error correction mechanisms like FEC and RTX
- Transport agnostic

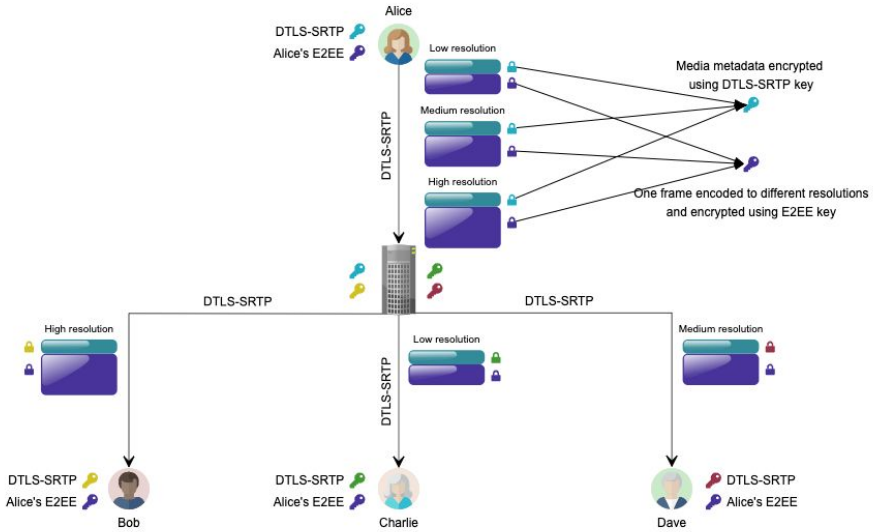
Conference Calls System Overview

- Endpoints sends multiple media streams to a central media server
- These streams are encrypted to the server HBH like DTLS-SRTP
- The server routes the streams to other endpoints in the call
- The server **has access** to the entire media contents



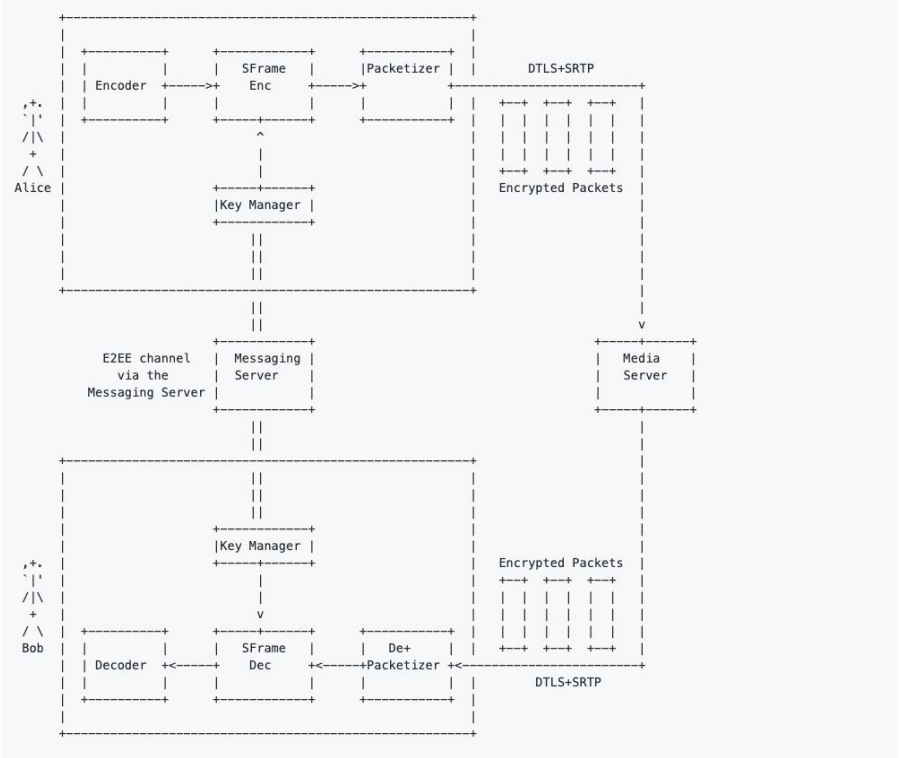
SFrame (Secure Frame)

- Mechanism to efficiently encrypt RTC traffic end to end
  - Encrypts the entire media frame rather than individual packets to minimize the overhead
  - Exposes only the metadata needed by the server to route the streams
  - Individual packets are still HBH encrypted
- SFrame keys are exchanged securely out of band between the endpoints
  - Each user has their own key to encrypt their outgoing traffic
  - Can be used with any KMS like Signal or MLS
  - Keys are exchanged via the signaling channel at the call setup and when the call participants changes
- The server can only access the media metadata



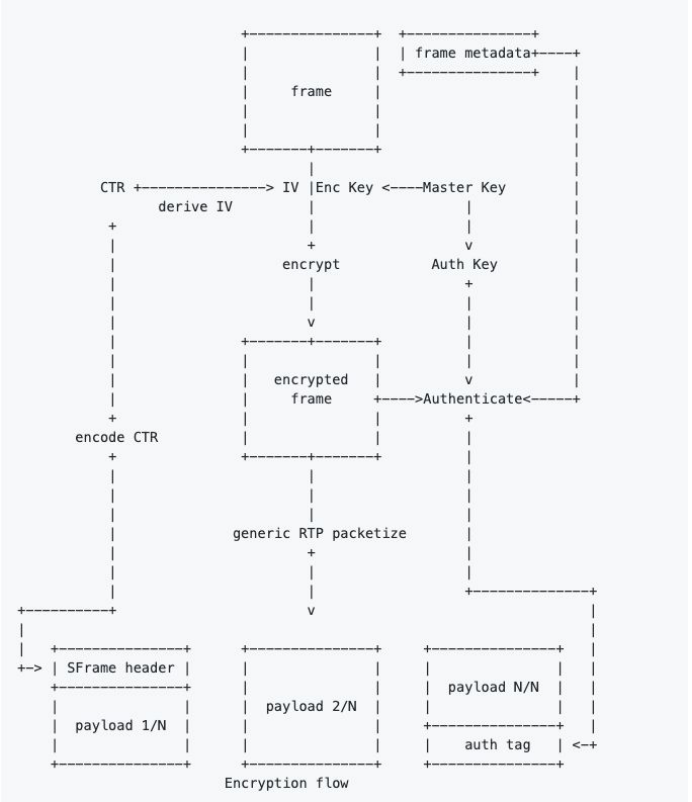
SFrame in WebRTC

- SFrame works with existing RTC frameworks like WebRTC
- The encryptor in injected after the frame is encoded and before it is packetized
- Media metadata are passed to the server using a special RTP header extension
- The server can construct the encrypted frame without access the contents

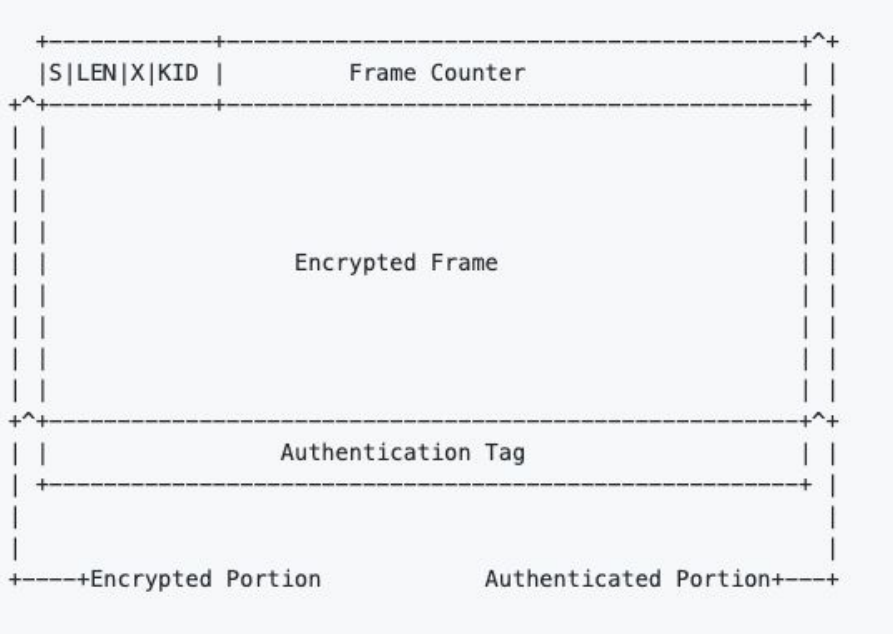


Encryption Schema

- Each endpoint creates and securely exchange their master key
- From the master key, SFrame derives 3 keys
  - Encryption key to encrypt the media frame
  - Authentication key to authenticate the encrypted frame. SFrame header and the media metadata
  - Salt key to derive the IV
- The entire payload is then split into smaller packets



Wire Format



SFrame payload

SFrame short header

SFrame long header

Encryption Overhead

- The encryption overhead mostly comes from the IV and authentication tag
- SFrame beats existing E2EE protocols because the overhead is amortized over the frame instead of per-packets
- SFrame also uses var-int encoding for the IV to reduce the overhead even more

Current Status

- Specs
  - SFrame draft
    - Mostly complete
    - Signature schema and keyID still WIP
  - Other documents needed
    - MLS-SFrame
      - KMS integration document
    - WebRTC-SFrame
      - The changes needed to WebRTC to support SFrame
        - Payload type
        - RTP metadata header ext
- Implementation
  - Implemented and launched in Google Duo since April 2019
  - Believe other implementations will be out soon

Next steps

- Are people interested in this?

- If Yes, where this should go ?
    i. New WG
    ii. Existing WG

Questions ?

Please submit your questions to

sframe@ietf.org