Encryption for content protection in streaming

SFrame WG - IETF 110
Dr Alex. Gouaillard
Two use cases - Two trust models

- Video Conference
  - Client trusted (special case of web apps, see youenn slides)
  - KMS trusted

Figure 1: Trusted and Untrusted Entities in PERC
Two use cases - Two trust models

- Streaming
  - Clients trusted (special case of webapp, see next slide)
  - KMS trusted

- Ingest link trusted
- Media platform trusted (need raw access for transcoding)

- Only delivery is encrypted (DRM)
- Encryption is media transport protocol specific.
Encrypted Media Extensions

W3C draft for playback protected content using the HTMLMediaElement.

The standard doesn’t specify the DRM subsystem itself but provides an API to interface/select a DRM subsystem.

Supported by almost all browsers using various DRM platforms: Widewine, Adobe DRM, PlayReady.
Using PlayReady and/or Widevine DRM Dynamic Common Encryption

VOD / Live → Assets → Storage → Streaming Endpoint

- DASH + PlayReady and/or Widevine
- HLS + PlayReady and/or FairPlay
- Smooth + PlayReady

Keys/licenses delivery service
- Token verification
- DRM licenses

Request DRM license → DRM license

Authenticate → Issue token

Players: Windows, Xbox, Apple, Android, Edge, Chrome
Two use cases - Two trust models

- Streaming
  - Ingest link trusted
    => this is changing, many social platforms have moved to RTMPs. It still requires that you trust the platform.

  - Media platform trusted (need raw access for transcoding)
    => many real-time platforms have a no-transcoding main path, and use simulcast or SVC codecs for adaptation.

Question 1: What if I do not trust the platform, and want to use my own keys?

Question 2: What if I want to use current DRM infrastructures with a different media transport, like webrtc?
From the sender perspective, with the current system, the platform must be trusted.

In the real time case, where ABR is done sender side, reusing the proposed SFrame + insertable frame + Native Key Management proposed by Apple would achieve the same content protection (as far as delivery is concerned) without needing to trust the platform.
Devil is in the detail

It’s likely more complicated than it seems.

There is the question of secure playback.

But there is something doable, and we will spend some time investigate anyway.

If anybody is interested, please join the effort.