

# SDP Security Assurance

draft-davis-valverde-srtp-assurance-00

Dispatch IETF 117

# Problem Statement

- A critical value in the SRTP cryptographic context is not signaled on the wire or shared between applications.
- The Rollover Counter (ROC) value is locally maintained and can very easily get out of sync in some very common scenarios
- The ROC is integral to the packet index used for to encrypt/decrypt SRTP packets.
- Out of Sync ROC has lead to many vendor interoperability issues, lost developer time and sour customer experiences.
- SRTP ROC is being handled in all SRTP Key Management protocols except SDP Security
- Ambiguity in RFC4568 about ROC handling, see errata 6291 and Scenario B, Problem 3

# Out of Sync ROC Scenarios

1. Joining ongoing broadcast session
2. Hold/Resume and Transfer (most common)
3. Application failovers without stateful syncs
4. Late addition of Secure SIPREC recording
5. Other “Improper” SRTP context resets
6. In general, RFC4568 had misconceptions about ‘there is no concept of a "late joiner" in SRTP security descriptions’
  - a. This failed to account for SBC/B2BUAs and other devices that may leverage SDP security to bridge and extensions sessions to new parties.

# Comparing Key Management Solutions

1. **SDP Security**: No ROC conveyance. See RFC4568
  - a. SRTPROCTxRate exists, unknown usage from RFC5159
2. **DTLS-SRTP/EKT-SRTP**: Convey ROC, SSRC. See RFC8870
3. **MIKEY**: Can convey ROC<sub>i</sub>, SSRC<sub>i</sub> as per RFC3830
  - a. Also RFC4771 for ROC in the SRTP packet
4. **ZRTP**: ROC conveyance unknown
5. **H.235.8**: Worked with the fact that a=crypto had no method of conveying ROC, SSRC force ROC to 0 always during negotiation.

# Why should we update SDP Security?

- SDP Security still used by many Enterprise (Endpoints, B2BUA, PBX, SBC, Conferencing, etc), Service Providers, and Cloud offerings (UCaaS, Conferencing, etc)
- Solution A or B is a small change that can provide a positive impact on many vendor interoperability scenarios
- This shouldn't inhibit natural growth/transition to DTLS-SRTP w/EKT or other next gen SRTP Key Management protocols

# Solution A: Reuse SDP Security Session Parameters

1. Leverage ability to convey session parameters via SDP Security's key=value
2. Register SSRC, ROC, SEQ as new postfix SRTP session parameters for a=crypto

```
a=crypto:<srtp-crypto-tag> <srtp-crypto-suite> <srtp-key-params> \  
  SSRC=<ssrc_value_hex> ROC=<roc_value_hex> SEQ=<last_known_tx_seq_hex> \  
  [<other-srtp-session-params>]
```

```
a=crypto:1 AEAD_AES_256_GCM \  
  inline:3/sxOxrbg3CVDrxeaNs91Vle+wW1RvT/zJWTCUNP1i6L45S9qcstjBv+eo0=\  
  |2^20|1:32 SSRC=0x00845FED ROC=0x0000 SEQ=0x0150
```

# Solution B: New SDP Attribute for key=value pairs

1. Convey SSRC, ROC, SEQ via a=srtpctx with a tag that matches the a=crypto value

```
a=srtpctx:<a-crypto-tag> ssrc=<ssrc_value_hex>;roc=<roc_value_hex>;seq=<last_known_tx_seq_hex>
```

```
c=IN IP4 192.0.0.1
m=audio 49170 RTP/SAVP 0
a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:d0RmdmcmVCspeEc3QGZiNWpVLFJhQX1cfHAwjSoj|2^20|1:32
a=crypto:2 AEAD_AES_256_GCM inline:HGAPy4Cedy/qumbZvpuCZSVT7rNDk8vG4TdUXp5hkyWqJCqiLRGab0KJy1g=
a=srtpctx:2 ssrc=0xBFBD; roc=0x0002; seq=0x3039
m=video 49172 RTP/SAVP 126
a=crypto:1 AEAD_AES_128_GCM inline:bQJXGzEPXJPClrd78xwALdaZDs/dLttBLfLE5Q==
a=srtpctx:1 ssrc=0xDD147C14; roc=0x0001; seq=0x50005
```