DTLS Security for SCTP


Magnus Westerlund
John Preuß Mattsson
Claudio Porfiri
3GPP Requested that IETF addressed the message size limitation in RFC 6083 (DTLS over SCTP)

TSVWG WG adopted draft-ietf-tsvwg-dtls-over-sctp-bis/ to address these issues

Last fall we found some security concerns with SCTP-Auth as well as realized some erroneous security assumptions

3GPP still need a solution that
   - Support larger messages sizes
   - Meets modern security requirements
   - Supports long lived SCTP associations

IPR Declarations exists:
Challenges with DTLS/SCTP

- In the development of draft-ietf-tsvwg-dtls-over-sctp-bis/ it has been realized that the current solution has some downsides
  - High requirements on DTLS implementations with several usages that aren’t common
    - Large DTLS record sizes beyond normal IP packet sizes
    - Usage of DTLS Connection-ID
  - Re-keying interacted with independent messages fragmentation and their placement in receiver buffer
    - Requires advanced tracking and SCTP stack interaction to guarantee delivery of DTLS message with a particular key
    - Fragile mechanism as any error will result in an SCTP Association termination
  - No Encryption Protection of SCTP protocol internals, only authentication through SCTP-AUTH
    - Current SCTP-AUTH does not protect against availability attack through reflection of sent packets
    - Dependent on updated SCTP-AUTH to prevent this combined with rekeying requirements
Lesson from SCTP-AUTH

- SCTP-AUTH consists of embedded authentication algorithms and external key management
  - SCTP-AUTH specific definition and implementation of cipher
  - Key requirements and usage are also SCTP-AUTH specific
  - Beyond the algorithm the specification and usage are not used somewhere else
- SCTP security is niche
  - Little review and research into security
  - Few implementations
- Therefore, we want to integrate security protocols as is
  - Benefit from a larger community work to maintain security protocol
  - Minimize integration aspects to keep areas of vulnerabilities small
DTLS in SCTP

- Alternative proposal to DTLS/SCTP written up in

- Trying to address the issues
  - Working on SCTP packet level instead of user messages
  - Defined as framework with
    - a general crypto chunk that carries the protected data, and performs negotiation of the mechanism using regular SCTP mechanism with downgrade protection
      - Similar integration to SCTP-AUTH
    - Protection engine specification that defines how to protect plain text, establish security contexts, and performs rekeying
    - Multiple protection engines may be specified, but we have defined a DTLS based one.
Crypto Chunk and Protection Engine

- Crypto Chunk Handler function as a dispatcher
  - Unprotected SCTP Packet payloads, i.e. a set of chunks are sent to Protection engine
  - SCTP Packet Payload returns as a crypto chunk with encrypted and integrity protection of the packet payload
  - Receiving an SCTP packet with a crypto chunk it is dispatched to the Protection Engine instance connected to the SCTP association
  - Protection engine returns unprotected payload if the integrity verification held else discard payload

- Protection Engine have two APIs
  - One to the SCTP stack that implements the Crypto chunk
  - One towards the ULP/Application where identity, certificates etc. are provisioned, and possibly out of band keying
Crypto Chunk State Machine

SCTP Association Init

1. Init includes Parameter negotiating Crypto Chunk support and which Protection Engines are supported
2. Peer respond (Init ACK) with the parameter indicating choice of common protection engine
3. Conclude handshake with Cookie Echo and Cookie ACK
4. Protection Pending: Protection Engine performs key-handshakes
5. When Security context is in place go to Protected state
6. Exchange PVALID chunk to verify that no Protection Engine downgrade happen
7. If Validation is okay move to established
DTLS Protection Engine

- Protection Engine defines how it uses the Crypto Chunk
- The unprotected payload are protected in a DTLS record
  - DTLS Replay Protection is used
  - DTLS record sequence number will be in strict SCTP packet transmission sequence order
- DTLS Handshake
  - Performed initial in Protection Pending by having the DTLS packetize its DTLS messages in crypto chunks and send them as SCTP packets to peer
  - DTLS responsible for retransmission of DTLS messages

- Interaction with SCTP Multihoming
  - Uses the SCTP current path
  - May require Heartbeat often enough to not timeout protection engine traffic
- DTLS Rekeying
  - During ongoing association Protection Engine creates a new DTLS connection and the handshake is sent as crypto chunk
    - When handshake is completed the endpoint switches to use the new
    - After a few RTTs to ensure that any SCTP packet with old key has been received the old DTLS connection is closed
Alternative realization per one of Michael Tüxen’s proposal

1. Send DTLS handshake, alerts etc as SCTP user messages
   - Use protection engine specific PPID for DTLS to identify these messages
2. Exchange with protection engine for crypto chunks:
   - To perform plaintext <-> encrypted transform
3. Same Protection engine to ULP interface
   • Benefit means that SCTP will hand retransmission, multihoming, and congestion control handled for handshakes
   • Downside is that SCTP stack must allow sending of DATA from DTLS, but not allow ULP before DTLS has completed validation so complicates SCTP stack API
   - Retransmission in DTLS SHOULD be turned off may impact generality of DTLS stacks
Kernel DTLS Implementation Concerns

- Michael raised a concern about the kernel <-> DTLS interaction as it can require either:
  - Kernel to User space call for each packet
  - Embedding/Loading the DTLS into kernel
- One possible way to minimize the impact is to split the DTLS implementation:
  - DTLS handshake stays in user land
  - DTLS record processing is implemented in kernel or even in hardware offload
  - Extensible used by Netflix and others
- From a spec perspective we will not be specific about these implementation choices
Benefits of DTLS in SCTP

- Encryption of all SCTP chunks, only common header and crypto chunk header is unencrypted
- Improved replay protection protecting also SCTP chunks without sequence numbers
- Rekeying is more robust as even an implementation that closes old DTLS connection immediately after establishing a new will at worst cause a bit of packet loss instead of SCTP association failure
- Rekeying not impacted by user message sizes used
- No need to track what user messages have left the sender buffer to know that rekeying can close old DTLS connection
- Uses DTLS 1.2 or 1.3 as it is defined
- Dependencies on DTLS features being implemented are much less
  - DTLS Connection ID
  - DTLS record sizes
  - Turning off Replay Protection
  - Not Use Key-Update
Path Forward

• Main Question to the WG:
  – Do we change solution path for DTLS protection of SCTP?

• Details of the proposal will continue to be discussed