Design Team Report on DTLS for SCTP

Design Team: Claudio Porfiri, Hannes Tschofenig, John Preuß Mattsson, Magnus Westerlund, Michael Tüxen, Tirumaleswar Reddy

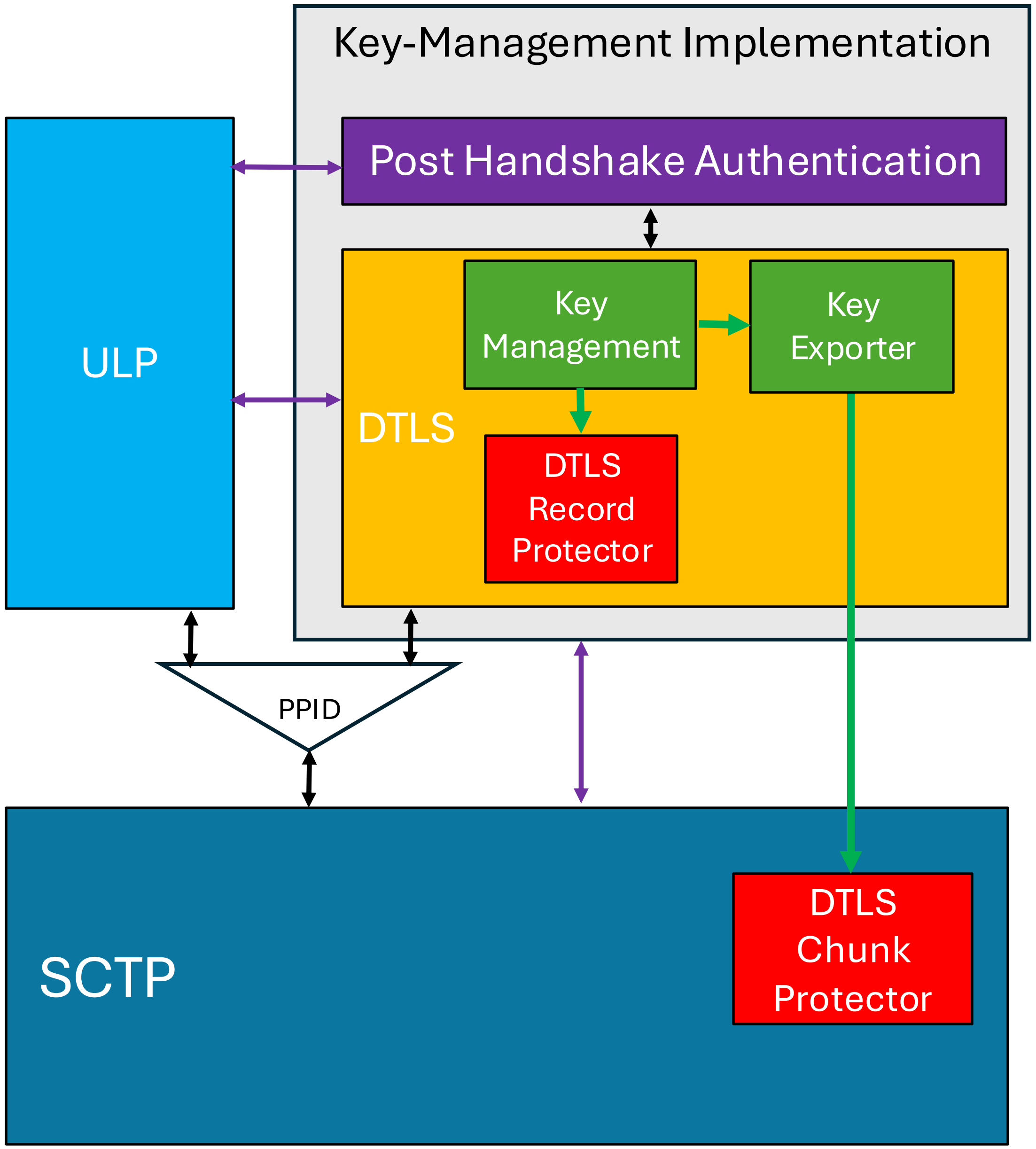
IETF-123 TSVWG WG

Introduction

- Design Team was tasked to see if a compromise could be reached
- The Design Team has succeeded in finding a proposal that is accepted compromise by the design team members

Proposal Overview

- DTLS Chunk Protector embedded in SCTP Stack
  - DTLS 1.3 Record protection
  - Replay, Encryption, Integrity
  - API to set keys etc.
- DTLS Key-Management
  - Use DTLS 1.3 Connection
    - Authentication
  - Use TLS Exporter to create keys for DTLS Chunk



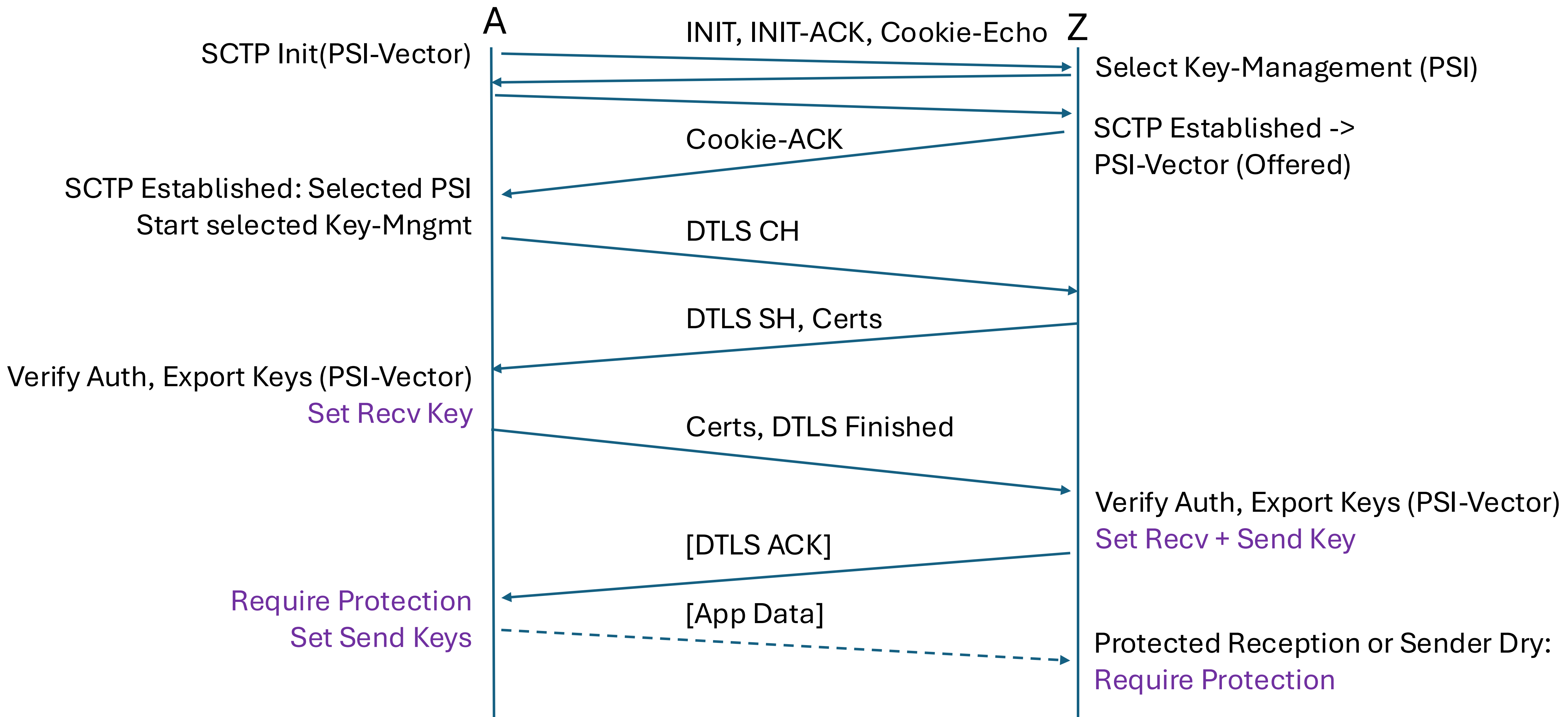
Design Principles

- DTLS Chunk
  - Be able to reuse the record processing in existing DTLS implementations for SCTP userland stacks
  - Have a clear and simple interaction model
    - With Key-Management
    - With SCTP stack
    - Support alternative key-management
  - Try to keep impact on SCTP stack low
  - Use Parameter so signal Key-Management solution (Protection Solution Indicators)

- Key-Management
  - Use a DTLS 1.3 Connection
  - Use RFC 9261 for mutual re-authentication
  - Use TLS Extended Key-Update to generate new master secrets to derive key from when rekeying
  - Key-Management Traffic is identified by own PPID
  - Downgrade protection is based on including Protection Solution identifiers in key-derivation
  - Messages exchanges using SCTP User Messages (in-order reliable)

Establishing Session Example

... : Plain Text
[...]: Protected by DTLS Chunk



Negotiating DTLS Chunk

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-------------------------------+-------------------------------+
|   Parameter Type = 0x80XX     |       Parameter Length        |
+-------------------------------+-------------------------------+
| Protection Solution #1        | Protection Solution #2        |
+-------------------------------+-------------------------------+
: Protection Solutions                                          :
+-------------------------------+-------------------------------+
| Protection Solution #N        | Padding                       |
+-------------------------------+-------------------------------+
```

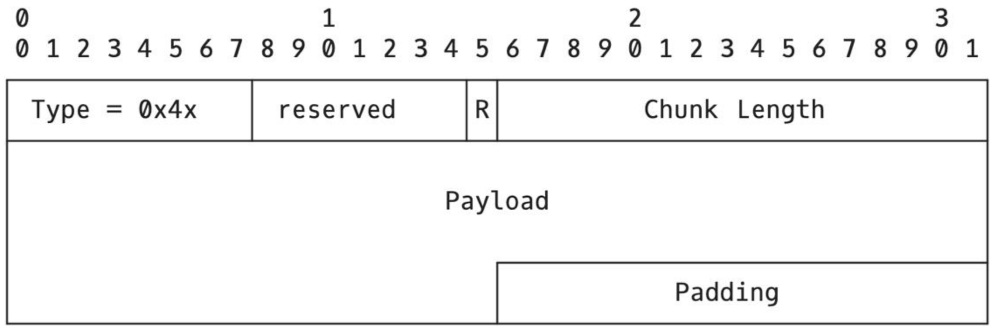
- Parameter indicating DTLS Chunk support exchanged in INIT and INIT-ACK
- INIT lists Protection Solutions that the initiator supports in decreasing preference order
- Protection Solutions
  - DTLS Chunk + Key-Management
  - Other Security Solutions
- Responder picks single solution in response

Downgrade Protection

- Goal: Prevent an attacker modifying the unencrypted parameters list of offerings
- Solution: Take the list of Protection Solutions from the INIT parameter (PSI-Vector)
  - Include it in both sides Key-Derivation using the TLS Exporter
- Key Derivation Requirement
  - Label indicating selected solution
  - Context: Include the PSI Vector

- Benefits
  - No extra message exchange to verify
  - Not vulnerable to weakest solution being possible to compromise
- Downside
  - Gives not explicit indication of downgrade attack simply fails the connection when peers have different sets of keys

DTLS Chunk

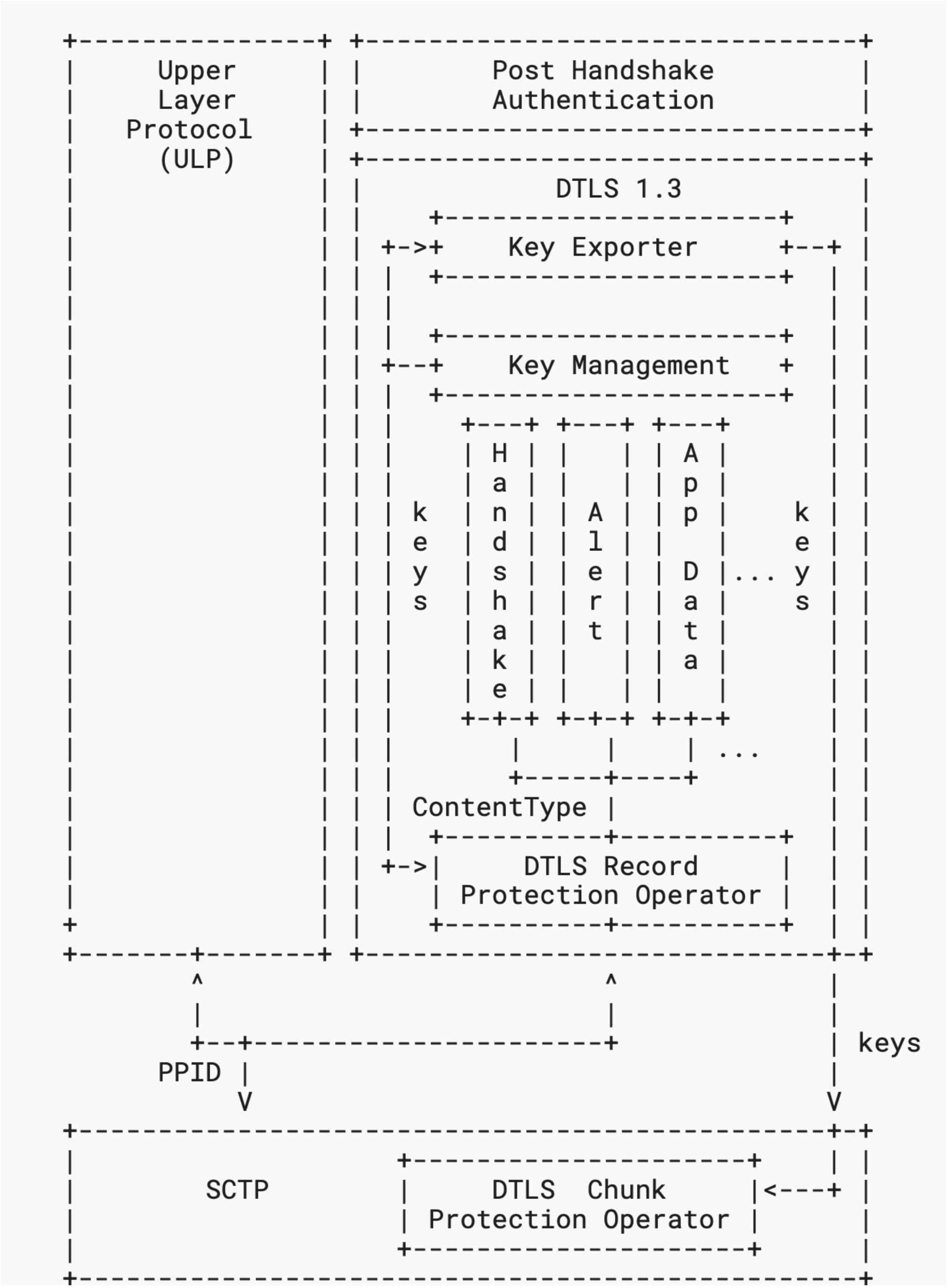



```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|0|0|1|C|S|L|E E|
+-+-+-+-+-+-+-+-+
| Connection ID |   Legend:
| (if any,      |
/  length as    /   C   - Connection ID (CID) present
|  negotiated)  |   S   - Sequence number length
+-+-+-+-+-+-+-+-+   L   - Length present
|  8 or 16 bit  |   E   - Epoch
|Sequence Number|
+-+-+-+-+-+-+-+-+
| 16 bit Length |
| (if present)  |
+-+-+-+-+-+-+-+-+
```

- DTLS Chunk is a SCTP Chunk
- Chunk Flag:
  - R: Restart Context Indicator else Primary Context used
- Payload: DTLS 1.3 DTLSChiphertext with DTLSInnerPlaintext.content being the protected SCTP Chunks
- DTLSChiphertext.unified_hdr:
  - Connection ID (if used)
  - Epoch
  - Sequence Number

DTLS Key-Management

- Responsible to:
    - Negotiate cipher suit to use:
        - DTLS Connection handshake
    - Perform Initial Key Establishment
        - DTLS Connection Handshake
        - Using TLS Exporter to create DTLS Key Context
    - Rekeying
        - TLS Extended Key-Update to establish new Forwards Secrecy master secret
        - Export New DTLS Chunk Key Context
    - Re-Authentication
        - Post Handshake Authentication (RFC 9261)

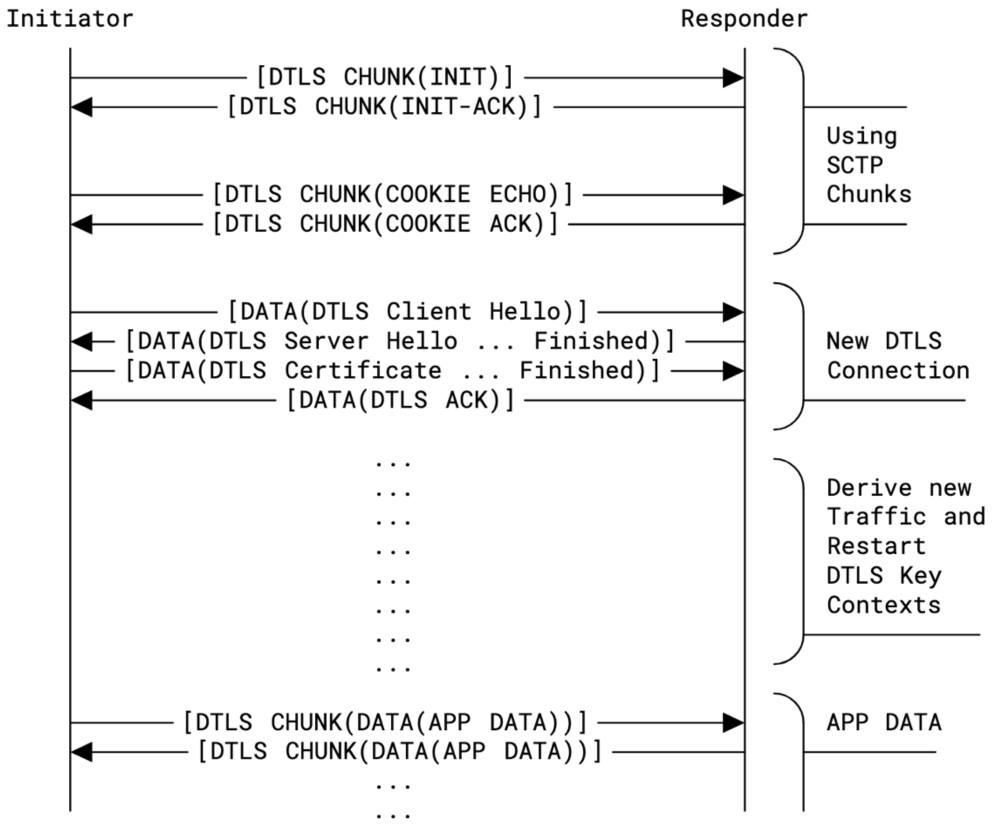


DTLS Key-Management

- Installing DTLS Chunk Key Contexts
  - Initially one can dead lock establishment:
    - Example: Installing send key before certain that peer can generate recv key, preventing working retransmissions
  - Key-Management need to define a robust procedure
    - Interaction between sent Key-Management Messages and procedure
- Rekeying Key Context Establishment
  - Extended Key-Update performed to completion by both side
  - Export new Keys + IVs
  - Establish and enable new Key Context (Epoch+1)
  - Drain network, delete old Key Context (Epoch)
- Re-authentication
  - Usage of RFC 9261 needs more details

SCTP Restart

- SCTP Restart enables re-establishment of SCTP association
- In use in mobile system to handle cases where the SCTP Stack is terminated but not the application protocol
- Prevention of hijacking and Denial of Service required
- Proposed change: INIT and INIT-ACK unprotected.



SCTP Restart

- DTLS Chunk Restart Key Context MUST be derived and installed by all endpoints.
- Enables all endpoints to respond to a SCTP Restart initiated by Peer

- Supporting Initiating SCTP Restart is a SHOULD
  - Requires storing Key+IVs for Restart Context persistent and securely
- This can’t be fulfilled or may not matter in some usages

Abstract API Context

- DTLS Chunk key-context ID:
  - SCTP Association
  - Primary or Restart (R) Context
  - DTLS Connection ID (if used)
  - DTLS Epoch
- Key Context State
  - Client Write Key + IV
  - Server Write Key + IV
  - Last used Sequence number (Sending)
  - Replay Window
  - Cipher suit to use
  - q, and v

Abstract API Functions

- Supported Cipher Suits by DTLS Chunk Implementation
- Set Key + IV + Cipher (Client/Server) in Key Context
- Destroy Key Context
- Select Context to use to Send
- Require Protected SCTP Packets in SCTP Association
- Get q: Read how many AEAD invocations done with key context
- Get v: Read number of failed AEAD Decryption invocations
- Set Replay Window Size

Socket API

- Will be defined in future version
- Based on FreeBSD and Linux developers

Individual Drafts

- DTLS Chunk:
  - https://datatracker.ietf.org/doc/draft-westerlund-tsvwg-sctp-dtls-chunk/
  - IPR declaration: https://datatracker.ietf.org/ipr/6220/

- DTLS Key-Management:
  - https://datatracker.ietf.org/doc/draft-tuexen-tsvwg-dtls-chunk-key-management/
  - No IPR Declaration

Next Steps

- Propose to adopt DTLS Chunk draft as WG item immediately
- Proceed to publication as soon as possible
  - DTLS Chunk has no dependencies on other documents
  - We have several smaller issues to resolve
- Propose to Adopt DTLS Key-Management (draft-tuexen-tsvwg-dtls-chunk-key-management)
  - Proceed towards publication at best possible speed
  - Dependent on https://datatracker.ietf.org/doc/html/draft-ietf-tls-extended-key-update-05
  - May published later than the DTLS Chunk