MLS Message Protection

Benjamin Beurdouche
benjamin.beurdouche@inria.fr

What’s in the draft today...

Initial draft for Message Protection.

A. Defines an Application Key Schedule (AKS)
   to go from the Epoch Secret to the Application messages encryption keys.

B. Defines which algorithms to use and on which objects
   to protect Application messages against active network attackers,
   with improved resistance to traffic analysis using optional padding.

Handshake Key Schedule

```
               init_secret_[n-1] (or 0)
                     |
                     V
update_secret -> HKDF-Extract = epoch_secret
                     |
                     +--> Derive-Secret(., "app", GroupState_[n])
                     |    = application_secret
                     |
                     V
               Derive-Secret(., "init", GroupState_[n])
                     |
                     V
               init_secret_[n]
```

Application Key Schedule

Two main ways of chaining secrets: interleaving or in parallel
independently from what we chain (Application secret or message encryption key)

- Group chaining of the secret (interleaving)
- Participant chaining of the secret (parallel)

Both would need anyway need to provide security properties such as:

- Forward Secrecy for secrets / messages
- PCS for secrets / messages

Application Key Schedule

Group chaining of the secret (interleaving)

+ Reduced complexity (storage)
+ Improved Forward Secrecy (no unused key stored for a long time)
- Reduced ability to handle out-of-order messaging for high frequency transmissions

Participant/Sender chaining of the secret (parallel)

+ Well-known design
+ Able to handle out-of-order messaging
- Higher complexity (storage)

Handshake Key Schedule

```
               init_secret_[n-1] (or 0)
                     |
                     V
update_secret -> HKDF-Extract = epoch_secret
                     |
                     +--> Derive-Secret(., "app", GroupState_[n])
                     |    = application_secret
                     |
                     V
               Derive-Secret(., "init", GroupState_[n])
                     |
                     V
               init_secret_[n]
```

Application Key Schedule

Derive a Participant Application Secret for each potential sender.

```
application_secret
         |
         V
Derive-Secret(., "app sender", [sender])
         |
         V
application_secret_[sender]_[0]
```

Application Key Schedule

Move forward the Participant Application Secret for each message.

```
application_secret_[sender]_[N-1]
          |
          +--> HKDF-Expand-Label(.,"nonce", "", nonce_length)
          |    = write_nonce_[sender]_[N-1]
          |
          +--> HKDF-Expand-Label(.,"key", "", key_length)
          |    = write_key_[sender]_[N-1]
          V
Derive-Secret(., "app upd","")
          |
          V
application_secret_[sender]_[N]
```

Application Message Protection

Expected Secrecy Properties

+ Forward Secrecy for each message inside a Participant chain
+ Forward Secrecy for each message across Group Updates
+ Post-Compromise Security for messages across Group Updates

Expected Authentication Properties

+ Weak Authentication from Group membership
+ Strong Authentication from Signatures (currently)

Application Message Protection

Signatures inside the ciphertext for Privacy reasons

```
struct {
    uint8  group[32];
    uint32 epoch;
    uint32 generation;
    uint32 sender;
    opaque content<0..2^32-1>;
} MLSSignatureContent;
```

Application Message Protection

Encryption via standard AEAD constructions and optional padding

```
struct {
    opaque content<0..2^32-1>;
    opaque signature<0..2^16-1>;
    uint8 zeros[length_of_padding];
} ApplicationPlaintext;

struct {
    uint8  group[32];
    uint32 epoch;
    uint32 generation;
    uint32 sender;
    opaque encrypted_content<0..2^32-1>;
} Application;
```

Summary

We have a reasonable initial draft for message protection

- Can we have PCS for all messages ? Do we want it ?
- We need some data to evaluate the computation and storage performance for the per-participant chaining and eventually the group chaining
- Should we keep signatures or use different constructions ?

We will need to have a look at...

- AES 128 vs AES 256
- AES GCM vs AES SIV
- PQ Primitives sizes