QUIC-MLS: MLS Integration at the Transport Layer

Benjamin Dowling	Kings College London
Britta Hale	NPS
Konrad Kohbrok	Phoenix R&D
Raphael Robert	Phoenix R&D
Xisen Tian	NPS
Bhagya Wimalasiri	Sheffield University

Transport Security: QUIC example

```
+------------+                               +------------+
|            |<---- Handshake Messages ----->|            |
|            |<- Validate 0-RTT Parameters ->|            |
|            |<--------- 0-RTT Keys ---------|            |
|    QUIC    |<------- Handshake Keys -------|    TLS     |
|            |<--------- 1-RTT Keys ---------|            |
|            |<------- Handshake Done -------|            |
+------------+                               +------------+
  |         ^
  | Protect | Protected
  v         | Packet
+------------+
|   QUIC     |
|  Packet    |
| Protection |
+------------+
```

Figure: Using TLS to Secure QUIC (RFC9001)

QUIC: *Interactive integrated key establishment, secure channel and transport protocol*

- Uses TLS 1.3 handshake and authenticated encryption (requires 1-RTT)
- 0-RTT transmission of encrypted data
- Forward secrecy with deterministic key updates: $\mathtt{secret}_{n+1} = \text{HKDF-Expand}(\mathtt{secret}_n, \mathtt{quic\ ku}, \emptyset, \text{H\_len})$
- Ubiquitous use in terrestrial internet apps

Some uses such as (deep)space would be optimized with handshake functionality support for...

- **Asynchronous communication, esp. at further distances**
- **Secure channels that can update/evolve without latency**
- **Good security guarantees even in low latency**
- **Handle delay**
- **Has analysis assumptions that match to the use**
- **Ability to support PQC**

QUIC is confined by....

- QUIC's TLS handshake **requires synchronous communication and 1RTT**
- **Interactive contribution *before* a secure channel**: Alice and Bob must both contribute to establishment of session key before encryption
- **High Risk in Low Latency**: 0-RTT mode has known issues (see RFC 8446)
- **Incentivizes bad security practice in space** e.g.,
    - long lived session keys
    - 0-RTT
- Is about to get a lot more bandwidth risk and packet loss risk to 1RTT under PQC
- **Key Update**: scales poorly and does not refresh randomness

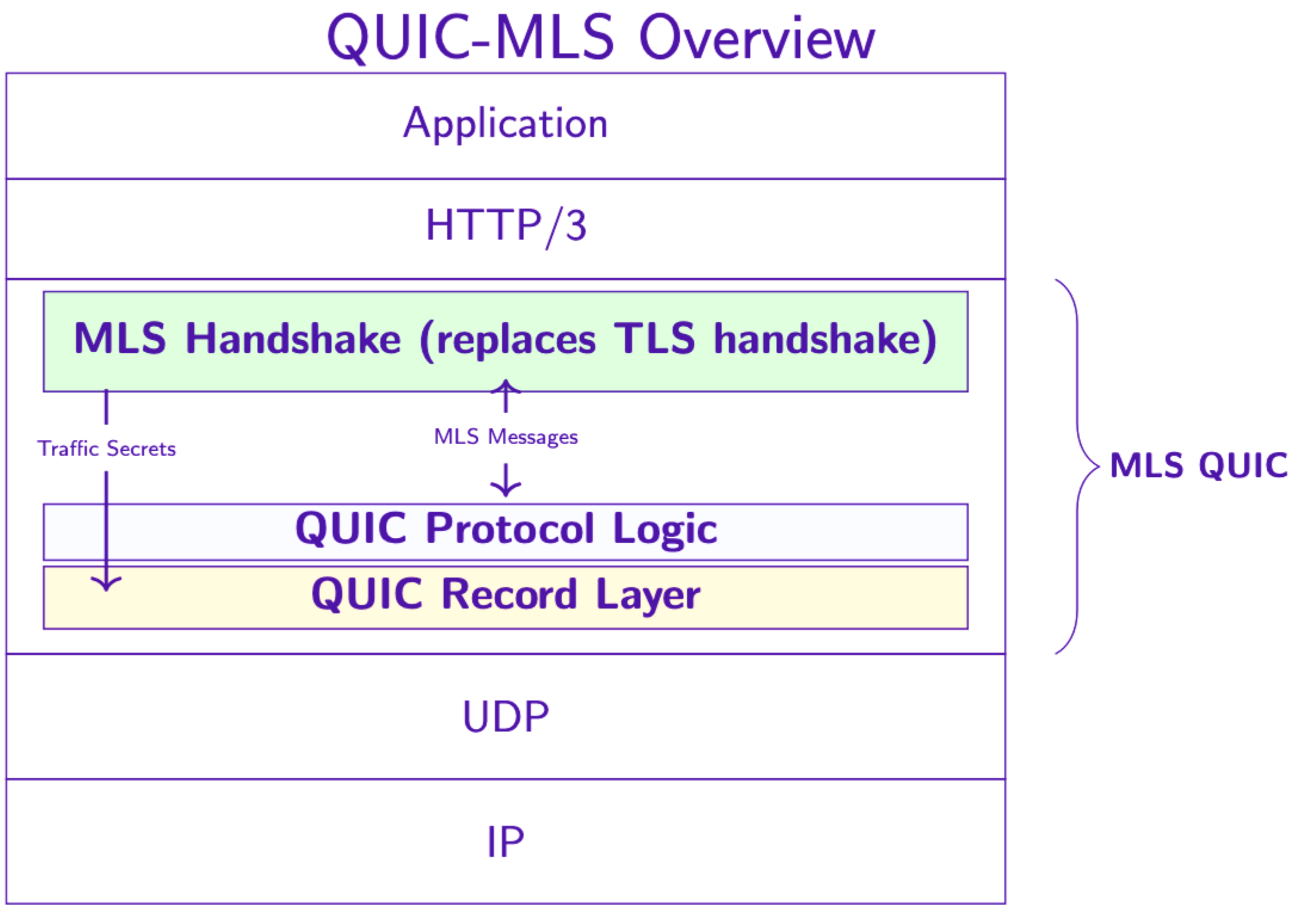
CKAs fit to these types of constraints.

MLS was intuitively designed for messaging and application layer use… BUT it is key establishment, and that is layer agnostic.

Just like how the TLS handshake is ‘slotted into’ QUIC, we can take that out and slot-in the MLS/RFC 9420 handshake, thus preserving all the other functionalities of QUIC while ensuring security suitable to the space use cases.

This concept could work for other network security protocols as well.

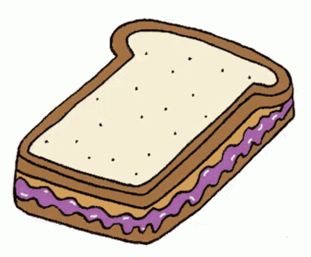
MLS Integration at a High Level



Integration from MLS Perspective

- <u>When</u>: When a protocol and its use-cases would benefit from qualities of MLS (security, scaling, etc.) AND integration is easier or better than retrofitting an existing protocol (e.g. TLS)
- <u>Goal</u>: Plug-and-play **do no harm** (to RFC9420) approach
- <u>How</u>: Use MLS Extensions (draft-ietf-mls-extensions)
  - MLS exporter key mechanism
  - Association of data with Commits
  - Binding of information to the Group Context

  Allows other layers above *and* below MLS (i.e. application, transport) to take advantage of MLS group state
- <u>What this Isn't</u>: Tunneling MLS in other protocols



QUIC-MLS Integration using MLS Extensions

MLS Extensions (draft-ietf-mls-extensions-07) can be used for other QUIC-HS requirements:

1. **Authentication of transport parameters** can be accomplished via attaching `quic_transport_parameters` to the Safe Additional Authenticated Data
2. **Application protocol agreement**: Content Advertisement Extension can be extended to ensure new group members support the same set of application protocols.
   - Content advertisement currently ensures group media type agreement

```
struct {
  ComponentID component_id;
  opaque aad_item_data<V>;
} SafeAADItem;

struct {
  SafeAADItem aad_items<V>;
} SafeAAD;
```

Safe Extensions AAD Field

```
struct {
    ComponentID component_id;
    opaque data<V>;
} ComponentData;

struct {
    ComponentData component_data<V>;
} AppDataDictionary;
```

Content Advertisement Application Data Dictionary

Summary

- MLS is an asynchronous protocol that is standardized, analyzed, and available for use today
- Common session-based protocols are synchronous and is non-optimal for some uses
- Key Recovery from disruptions requires additional handshakes under session-based protocols. It requires no handshake interaction in MLS.
- Security
  - Post Compromise Security is designed into MLS and is increasingly becoming an expected property for secure channels.
  - Post Quantum performance can be optimized with MLS
- MLS Extensions provide useful APIs for integration (e.g. rather than tunneling with MLS Application Messages)

Contact

- Benjamin Dowling benjamin.dowling@kcl.ac.uk
- Britta Hale britta.hale@nps.edu
- Konrad Kohbrok konrad.kohbrok@datashrine.de
- Raphael Robert ietf@raphaelrobert.com
- Xisen Tian xisen.tian1@nps.edu
- Bhagya Wimalasiri b.m.wimalasiri@sheffield.ac.uk