Toward TLS 1.3 in Mbed TLS

Hanno Becker & Hannes Tschofenig
November 3, 2020
Current and future TLS-based standards

• TLS is long standardized – when is Mbed TLS going to support it?
• DTLS is close to standardization
• cTLS is in development
• QUIC uses TLS handshake logic

Questions:
• (When/How) Do we want to support those standards in Mbed TLS?
• How do we accommodate the large changes and variations across the stack these protocols bring, without fragmenting the code too much?
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
- cTLS in development
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
- cTLS in development
- QUIC used TLS 1.3 handshake logic
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
- cTLS in development
- QUIC used TLS 1.3 handshake logic

Questions:
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
- cTLS in development
- QUIC used TLS 1.3 handshake logic

Questions:
- (When/How) Do we want to support those standards in Mbed TLS?
Current and future TLS-based standards

- TLS 1.3 is long standardized – when is Mbed TLS going to support it?
- DTLS 1.3 getting close to standardization
- cTLS in development
- QUIC used TLS 1.3 handshake logic

Questions:
- (When/How) Do we want to support those standards in Mbed TLS?
- How do we accommodate the large changes and variations across the stack these protocols bring, without fragmenting the code too much?
Work in progress: TLS 1.3 Prototype + MPS

- TLS one.pnum/three.pnum prototype under development: h/t_t.ligaps://github.com/hannestschofenig/mbedtls/tree/tls/one.pnum/three.pnum-prototype
- Most functionality is available, but code isn’t production quality yet.
- We’re working on cleaning up the code and upstreaming one piece at a time.
- In parallel, have worked on replacement of messaging layer of Mbed TLS ('MPS') which will facilitate the integration of the above standards beyond TLS one.pnum/three.pnum.
Work in progress: TLS 1.3 Prototype + MPS

• TLS 1.3 prototype under development:
  https://github.com/hannestschofenig/mbedtls/tree/tls13-prototype
Work in progress: TLS 1.3 Prototype + MPS

• TLS 1.3 prototype under development: 
  https://github.com/hannestschofenig/mbedtls/tree/tls13-prototype

• Most functionality is available, but code isn’t production quality yet.
Work in progress: TLS 1.3 Prototype + MPS

- TLS 1.3 prototype under development: https://github.com/hannestschofenig/mbedtls/tree/tls13-prototype
- Most functionality is available, but code isn't production quality yet.
- We’re working on cleaning up the code and upstreaming one piece a time.
Work in progress: TLS 1.3 Prototype + MPS

- TLS 1.3 prototype under development:
  https://github.com/hannestschofenig/mbedtls/tree/tls13-prototype

- Most functionality is available, but code isn't production quality yet.

- We’re working on cleaning up the code and upstreaming one piece a time.

- In parallel, have worked on replacement of messaging layer of Mbed TLS ('MPS') which will facilitate the integration of the above standards beyond TLS 1.3.