TLS 1.3: What has changed

Dmitry Belyavskiy

Cryptocom

History

- SSLv2 – 1995
- SSLv3 – 1996
- TLS1.0 – RFC 2246, 1999
- TLS1.1 – RFC 4346, 2006
- TLS1.2 – RFC 5246, 2008
- TLS1.3 – RFC ????, 2018

TLS 1.3 design goals

- Better traffic protection
- Remove all unsafe stuff
- Make TLS faster

TLS 1.3: Handshake redesign

ClientHello

ServerHello

Certificate

ServerKeyExchange

ServerHelloDone

ClientKeyExchange

[ChangeCipherSpec]

Finished

[ChangeCipherSpec]

Finished

Application Data

ClientHello

+key_share

ServerHello

+key_share

Certificate

Finished

{Application Data}

Finished

Application Data

TLS 1.3 and DPI

- Before:
  non-encrypted SNI
  non-encrypted Certificate
- Now:
  non-encrypted server_name
  encrypted Certificate
- Future:
  encrypted SNI

- Certificate-based DPI is not applicable!

Faster content delivery

- Before TLS 1.3: 2 Round-trips
- TLS 1.3: 1 Round-trip
- Make CDNs happy

Authentication and certificates

- RSA-PSS instead of PKCS1-v1.5
  - Avoid Bleichenbacher attacks
- No more DSA certificates
- No more static DH
- Brand-new PSK mode

Battle against surveillance

- PFS is mandatory
- Tries to re-enable unsafe methods:
  - ➢Return back RSA key exchange
  - ➢Allow repeat Diffie-Hellman random data
  - ➢3-side protocol variations
- Motivation: debugging purposes
- Not accepted by community, to be continued

Cipher modes

- AES-128, AES-256, ChaCha
- AEAD modes only:
  - AES-GCM, AES-CCM, ChaCha-Poly1305
- No more:
  - CBC modes
  - DES/3DES, RC4, ARIA, CAMELLIA
  - SHA1, MD5...
- No more compression

Ciphersuites Redesign

- Before:
  Ciphersuite = Key Exchange + Authentication + Cipher + MAC + PRF
- After:
  Key Exchange + Authentication
  AEAD-based cipher
  HKDF instead of PRF

0-RTT mode

- Reuse previously established keys
- NO perfect forward secrecy
- Vulnerable to replay attack

Problem: Middleboxes

- Try to simplify handshake
- Middleboxes do not recognize TLS 1.3
- Redesign: make handshake more similar to previous handshake versions

Support

- OpenSSL 1.1.1 (when available)
- Mozilla Firefox, Google Chrome
- Cloudflare, Akamai...

Conclusions

- Almost brand-new
- The safest TLS protocol
- The fastest TLS protocol
- Waiting for encrypted SNI
- Waiting for DTLS

- Waiting for Russian GOST :)

Questions?

- beldmit@cryptocom.ru