Real-world post-quantum TLS in Wireshark

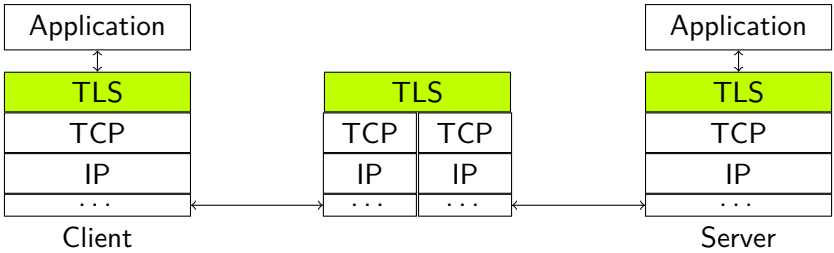
Wednesday November 1st, 2023

Peter Wu
Wireshark Core Developer
peter@lekensteyn.nl

About me

- Wireshark contributor since 2013, core developer since 2015.
- Areas of interest: TLS, QUIC, HTTP/3, Lua, security, ...
- Cloudflare Research team. Recently worked on rolling out post-quantum TLS.

- Standard for securing network traffic. Web (HTTP), e-mail, databases, etc.
- Provides secure communication channel between two endpoints (client and server).
- Network protocol with two components:
  - Handshake Protocol: exchange capabilities, establish trust and establish keys.
  - Record Protocol: carries messages and protects application data fragments.



- Powerful quantum computers are expected in 15 to 40 years.[1]
- Essentially all Internet traffic today can be decrypted by these.
- Post-quantum (PQ) cryptography was designed to be secure against this threat.
- In active development: US National Institute of Standards and Technology (NIST) is almost done standardizing the initial post-quantum public-key algorithms.

[1] `https://blog.cloudflare.com/post-quantum-for-all/`

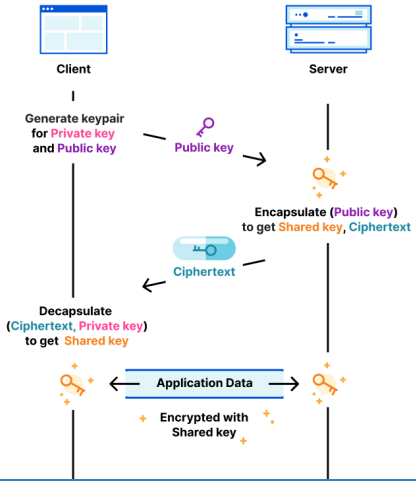
- Symmetric encryption: sender and receiver have the same secret key.
- Authenticated Encryption with Additional Data (**AEAD**) added in TLS 1.2: AES-GCM, ChaCha20-Poly1305.
- Legacy (TLS $\leq$1.2): combine ciphers such as AES-CBC or RC4 with a Hashed Message Authentication Code (HMAC): HMAC-SHA256, HMAC-SHA1.
- Modern symmetric encryption is already post-quantum secure.

- Public-key cryptography: different private and public key. Private encryption/signing key. Public decryption/verification key.
- Digital signature algorithms: RSA, **ECDSA**.
- Key agreement or key exchange (KEX): RSA, **ECDHE** (Elliptic Curve Diffie-Hellman with ephemeral keys).
- Classical signature and key agreement algorithms are not PQ-secure.

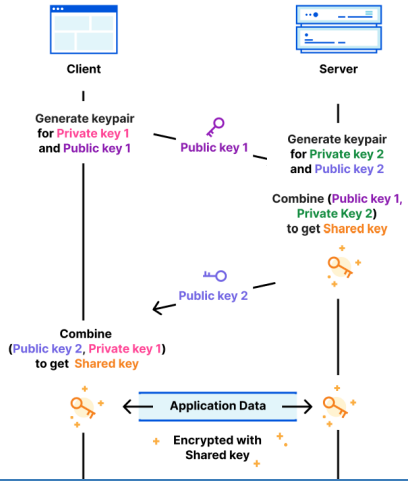
		Size (bytes)		CPU time (lower is better)	
	PQ	**Public key**	**Signature**	**Signing**	**Verification**
Ed25519	❌	32	64	1 (baseline)	1 (baseline)
RSA-2048	❌	256	256	70	0.3
Dilithium2	✅	1,312	2,420	4.8	0.5
Falcon512	✅	897	666	8*	0.5
SPHINCS$^+$128s	✅	32	7,856	8,000	2.8
SPHINCS$^+$128f	✅	32	17,088	550	7

Source: `https://blog.cloudflare.com/nist-post-quantum-surprise/` (2022)

Key Encapsulation Mechanism (KEM)



Diffie–Hellman (DH)



- Hybrid key agreement: Combine shared secrets from classic ECDHE (X25519) and post-quantum Kyber768 draft version.
- At least as secure as current X25519 deployments.
- Kyber is the basis for the future NIST FIPS 203 standard, *Module-Lattice-Based Key-Encapsulation Mechanism* (ML-KEM)[2].

[2] Initial Public Draft: `https://csrc.nist.gov/pubs/fips/203/ipd` (2023)

Group name	Public key size Client	Public key size Server	CPU time Client	CPU time Server
ECDHE: X25519	32		1 (baseline)	
ECDHE: NIST P-256	65		3.25	
ECDHE: NIST P-384	97		50.4	
ECDHE: NIST P-521	133		116.7	
PQ: Kyber768	1184	1088	5.53	3.53
Hybrid: X25519Kyber768Draft00	1216	1120	6.53	4.53

- Lower CPU time is better.
- Note: optimized Kyber768 versions are even faster than P-256.

Group name	Group ID	Public key size Client	Server
X25519	29, 0x001d	32	
NIST P-256	23, 0x0017	65	
NIST P-384	24, 0x0018	97	
NIST P-521	25, 0x0019	133	
X25519Kyber768Draft00	25497, 0x6399	1216	1120

```
▾ Extension: key_share (len=1263) X25519Kyber768Draft00, x25519
    Type: key_share (51)
    Length: 1263
  ▾ Key Share extension
      Client Key Share Length: 1261
    ▸ Key Share Entry: Group: Reserved (GREASE), Key Exchange length: 1
    ▾ Key Share Entry: Group: X25519Kyber768Draft00, Key Exchange length: 1216
        Group: X25519Kyber768Draft00 (25497)
        Key Exchange Length: 1216
        Key Exchange: 91299366af91cdb945067ccd9ee60bdae028af3fc8dc7bea823930946
    ▸ Key Share Entry: Group: x25519, Key Exchange length: 32
```

- Google Chrome: enable *TLS 1.3 hybridized Kyber support* at `chrome://flags/#enable-tls13-kyber`.
- Open `https://pq.cloudflareresearch.com/`
- Cloudflare enabled PQ KEX in 2022 (about 20% Internet)
- Google enabled support server-side in 2023.
- Browsers:
  - Google Chrome enabled for 1% in 2023.
  - Mozilla Firefox is expected in 2024.
- `https://lekensteyn.nl/files/captures/chromium119-dsb.pcapng`

What to look for

- Locate Client and Server Hello messages: `tls.handshake.type in {1, 2}`
- For PQ KEX, both client and server TLS extensions must have:
  - Supported Versions with TLS 1.3.
  - Supported Groups with X25519Kyber768Draft00 (25497).
  - Key Shares with X25519Kyber768Draft00.
- QUIC: runs over UDP instead of TCP. Uses TLS 1.3 for security.
- Match TLS Server Name with: `tls.handshake.extensions_server_name`
- Tip: use stream index for linking packets via Custom column:
  - `tcp.stream or quic.connection.number or udp.stream`

- With Cloudflare Tunnel you can securely expose a server sitting within an internal network to the Internet by running the cloudflared service next to it.
- Uses HTTP/2 over TLS, but there also QUIC with experimental PQ support[3].
- Example: `cloudflared tunnel --hello-world --post-quantum`
- Dump secrets with a debugger or compile Go code with tls.Config#KeyLogWriter.
- `https://lekensteyn.nl/files/captures/cloudflared-quic-pq-dsb.pcapng`

[3] `https://blog.cloudflare.com/post-quantum-tunnel/`

- Client or server were not properly configured with PQ support.
- TLS 1.3 is not enabled or TLS 1.2 or older is forced.
- The wrong server software was targeted by the client.
- An intercepting TLS middlebox was in use that did not support PQ.

- Maximum Transmission Unit (MTU): typically 1500 for Ethernet. Can be lower due to tunneling/VPN overhead.
- Client connects, but during the TLS handshake times out waiting for the server.
- Client capture shows that the TCP handshake succeeds, but
  - Case 1: TLS Client Hello is sent, but never ACKed.
  - Case 2: TLS Server Hello is partially returned.[4] Check TCP sequence numbers.
- Solution: reduce MTU or apply TCP Maximum Segment Size (MSS) clamping.

[4] `https://lekensteyn.nl/files/captures/tls-server-mtu-issue.pcap`

- If a TLS 1.3 server prefers a different key exchange group, it can send a Hello Retry Request (HRR).
- Client receives a TLS alert (Illegal Parameter) during the TLS handshake.
- Affects servers written in the Rust programming language using rustls.[5]
- Fixed in rustls 0.20.9 and 0.21.7 (August 2023).
- Servers must copy client Session ID into HRR to simulate TLS 1.2 session resumption for *middlebox compatibility mode*.
- `https://lekensteyn.nl/files/captures/time-hrr-rustls-bug.pcapng`

[5] `https://github.com/rustls/rustls/issues/1424`

- Cloudflare's reverse proxy to origin servers support PQ.[6]
- It can directly send the PQ key share ("preferred mode").
- Or advertise PQ support, but initially send X25519 ("supported mode").
- The latter can trigger a Hello Retry Request to ask the client to retry with the PQ key share. Adds one extra roundtrip.
- `https://lekensteyn.nl/files/captures/pq-origin-dsb.pcapng`

[6] `https://blog.cloudflare.com/post-quantum-to-origins/`

- Post-quantum cryptography is here to protect data in the future.
- Use a key log file to enable TLS decryption in Wireshark.
- Embed these secrets in a pcapng file for easier distribution.
- Use the latest Wireshark version for the best results.
- For a more detailed background and key extraction from other applications, see `https://lekensteyn.nl/files/wireshark-ssl-tls-decryption-secrets-sharkfest18eu.pdf`

peter@lekensteyn.nl
lekensteyn.nl
@Lekensteyn@infosec.exchange
@Lekensteyn