Preface

- This is definitely not a complete overview
- We will mainly focus on the authentication part of TLS 1.3
- We’re not pitching a draft here or any path to go:
  - Opening the door for discussion
  - Opening the door for experimental design

Brief note on the KEX of TLS 1.3

- KEX: Key Exchange
- First NIST KEM to-be standard:
  - Kyber
- Round 4 of KEMs:
  - SIKE, BIKE, HQC
- TLSWGs hybrid mechanism: draft-ietf-tls-hybrid-design
  - Parked for the moment
  - The way to go

¹ if NIST can’t resolve the patent situation, they say they may still go for NTRU
Authentication in TLS 1.3

- Certificate-based authentication
- Pre-shared key
- Password-based authentication
Authentication in TLS 1.3

- Certificate-based authentication
- Pre-shared key
- Password-based authentication
Certificate-based authentication

- Usage of signatures:
  - **Online** signatures:
    - Signature of the handshake: signing and verifying
  - **Semi-online** signatures (signed at different moments, and verified by different parties)
    - Signature(s) of the certificate chain: offline signing and online (offline) verifying
    - OSCP staple: offline signing and online (offline) verifying
      - Online (i.e. OCSP and CRL) checks are not, generally, performed by major browsers
      - Underlying system certificate library performs the checks
    - SCT: offline signing and online (offline) verifying
      - Depends on browsers policy:
        - Google Chrome requires CT log inclusion
        - Safari requires a varying number of SCTs ([https://support.apple.com/en-gb/HT205280](https://support.apple.com/en-gb/HT205280))
        - Firefox or Brave do not check or require the use of CT logs ([https://bugzilla.mozilla.org/show_bug.cgi?id=1281469](https://bugzilla.mozilla.org/show_bug.cgi?id=1281469))
# Post-quantum signatures: tradeoffs

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Public key bytes</th>
<th>Signature bytes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA-2048</td>
<td>272</td>
<td>256</td>
<td>Pre-quantum</td>
</tr>
<tr>
<td>Ed25519</td>
<td>32</td>
<td>64</td>
<td>Pre-quantum</td>
</tr>
<tr>
<td>Dilithium-II (MLWE)</td>
<td>1312</td>
<td>2420</td>
<td>NIST’s “primary” selection/recommendation</td>
</tr>
<tr>
<td>Falcon-512 (NTRU)</td>
<td>897</td>
<td>666</td>
<td>NIST's choice for small signatures “if implemented correctly”</td>
</tr>
<tr>
<td>SPHINCS+ 128s</td>
<td>32</td>
<td>8080</td>
<td>slow, conservative</td>
</tr>
<tr>
<td>XMSS (RFC8391)</td>
<td>32</td>
<td>979</td>
<td>Stateful hashing not fit for general purpose</td>
</tr>
<tr>
<td>On-ramp candidate</td>
<td>??? UOV: &gt;400k uncompressed</td>
<td>“small and fast to verify” UOV: smaller than RSA</td>
<td>Probably no standards before 2028</td>
</tr>
</tbody>
</table>
Prior work: PQ (experiments) on the web

- Google/Cloudflare: CECPQ1, CECPQ2 key exchange
  - [https://www.imperialviolet.org/2018/12/12/cecpq2.html](https://www.imperialviolet.org/2018/12/12/cecpq2.html)

- Cloudflare: Performance impact of large certificate chains

- OpenSSH 8.9 uses NTRUPrime as default key exchange algorithm
Prior work: Academic studies

- PQ authentication in TLS: a performance study
  - https://eprint.iacr.org/2020/071

- PQ TLS on embedded platforms
  - https://eprint.iacr.org/2020/308 mbedTLS: Kyber + SPHINCS+

- Prototyping PQ KEX and authentication in OpenSSL and OpenSSH
  - https://eprint.iacr.org/2019/858

- Post-Quantum password-based authentication using RLWE
  - https://eprint.iacr.org/2017/1192
Selection of ongoing IETF work

- [pqc@ietf.org](mailto:pqc@ietf.org)
- **TLS:**
  - [draft-ietf-tls-hybrid-design](https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/)
  - [draft-celi-wiggers-tls-authkem](https://datatracker.ietf.org/doc/draft-celi-wiggers-tls-authkem/)
- **CFRG:**
  - [XMSS / LMS](https://datatracker.ietf.org/doc/html/rfc7293/) RFCs
- **LAMPS:**
THANK YOU!

@claucece
@thomwiggers

See also our CFRG slides for more links!