
THE CHALLENGE OF THE PQ MIGRATION AND KEM-BASED AUTHENTICATION

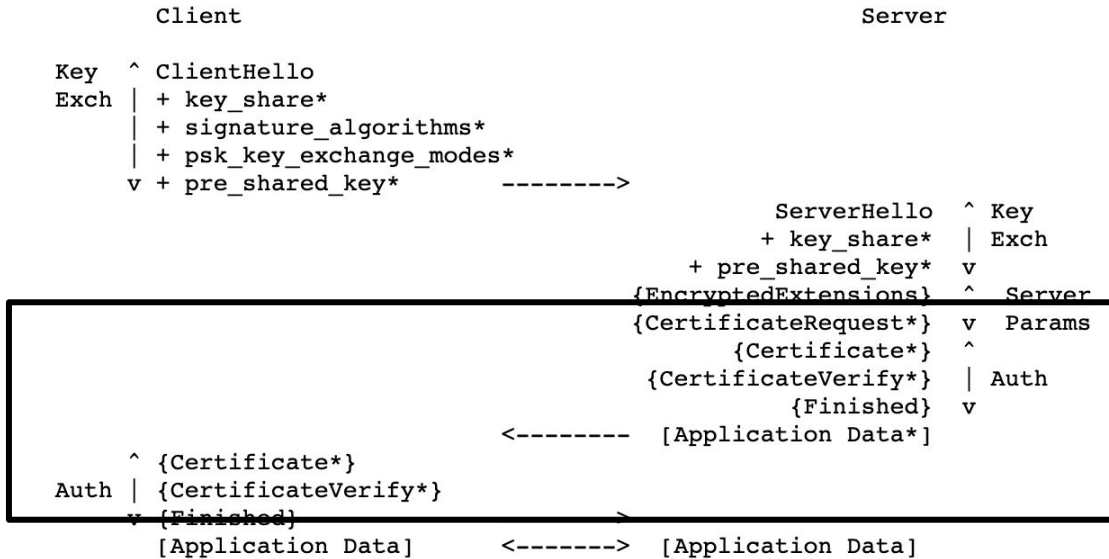
Sofía Celi

SPOILERS

- This talk is going to be about *authentication*
- This talk is not going to touch key exchange phase of protocols

LET'S START BRIEFLY WITH TLS 1.3

Figure 1 below shows the basic full TLS handshake:



+ Indicates noteworthy extensions sent in the previously noted message.

* Indicates optional or situation-dependent messages/extensions that are not always sent.

{ } Indicates messages protected using keys derived from a [sender]_handshake_traffic_secret.

[] Indicates messages protected using keys derived from [sender]_application_traffic_secret_N.

Figure 1: Message Flow for Full TLS Handshake

From:

<https://datatracker.ietf.org/doc/html/rfc8446>

TLS 1.3

- Two parts:
 - **A handshake protocol**
 - A record protocol
- Properties:
 - **Authentication** (certificates, password, pre-shared key)
 - Post-handshake authentication
 - Confidentiality
 - Integrity

CERTIFICATE-BASED AUTHENTICATION

SIGNATURES

- **Authentication:** Assurance that you are talking with the correct party
 - Party signs with the private key
 - Peer (or anyone) verifies with the public key
- Server-only authentication or mutual (both server and client) authentication
- Need to verify the used public key

SIGNATURE ALGORITHMS

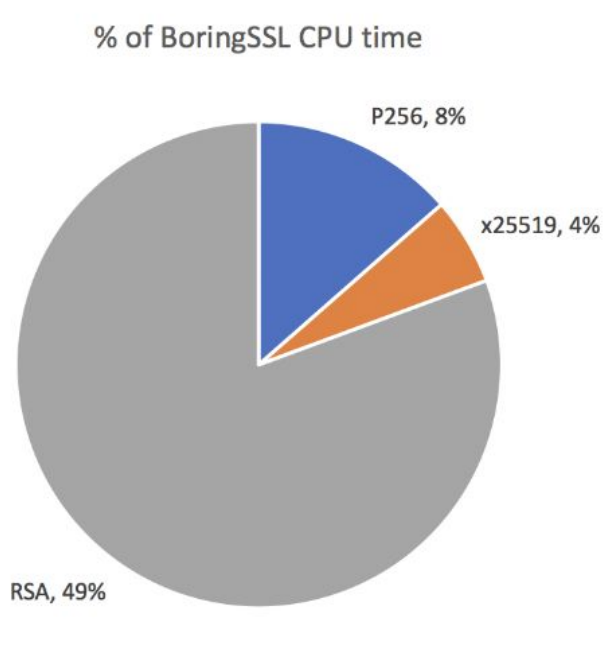
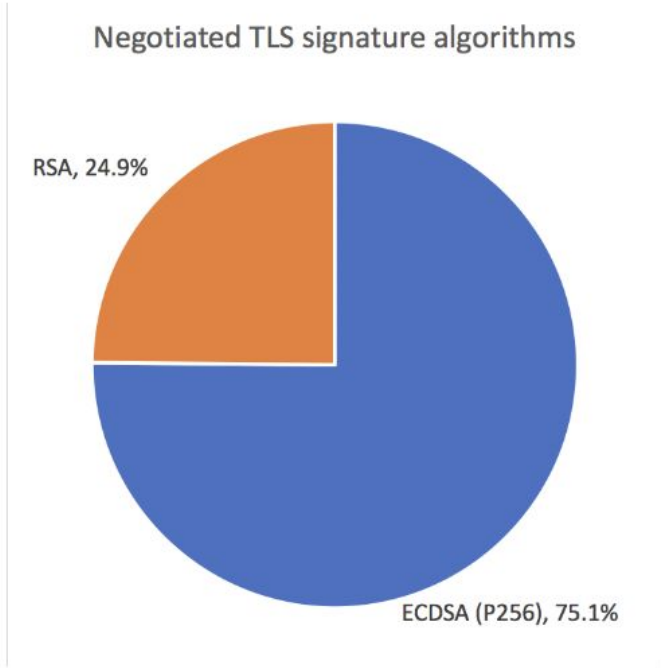
Scheme Name	Problem	Public key size (bytes)	Signature size (bytes)
RSA-2048	Factoring	272	256
Ed25519	Elliptic curve discrete logarithm	32	32
Dilithium2	Lattice-based (MLWE/MSIS)	1312	2420
Falcon-512	Lattice-based (NTRU)	897	690
Rainbow-I-Classic	Multi-variate equations	161,600	66
XMSS	Hash-based	32	979
SQISign (6983)	Isogeny-based	64	204
MAYO	Multivariate Quadratic	830	420

SIGNATURE ALGORITHMS

Scheme Name	Problem	Public key size (bytes)	Signature size (bytes)
RSA-2048	Factoring	272	256
Ed25519	Elliptic curve discrete logarithm	32	32
Dilithium2	Lattice-based (MLWE/MSIS)	1312	2420
Falcon-512	Lattice-based (NTRU)	897	690
Rainbow-I-Classic	Multivariate equations	161,600	66
XMSS	Hash-based	32	979
SQISign (6983)	Isogeny-based	64	204
MAYO	Multivariate Quadratic	830	420

SIGNATURE ALGORITHMS

Scheme Name	Signing (cycles)	Verification (cycles)	On
Dilithium2	2348703	529106	Intel Core-i7 6600U (Skylake) CPU
SQISign	6,673 million cycles	82 million cycles	Intel Core i7-6700 clocked at 3.40 GHz with turbo-boost deactivated
MAYO	2.50 million cycles	1.3 million cycles	Intel i5-8400H CPU



'How expensive is crypto anyway?' (2017):

<https://blog.cloudflare.com/how-expensive-is-crypto-anyway/>

Profiled server in CF Frankfurt data center: 2 Xeon Silver 4116 processors

- Latests results of:
 - SQSign: De Feo, Leroux, Wesolowski, 'New algorithms for the Deuring correspondence: SQISign twice as fast', De Feo <https://eprint.iacr.org/2022/234.pdf>
 - XMSS: Kampanakis, Fluhrer, 'LMS vs XMSS: Comparison of two Hash-Based Signature Standards': <https://eprint.iacr.org/2017/349.pdf>
 - Breaking Rainbow: Beullens, 'Breaking Rainbow Takes a Weekend on a Laptop': <https://eprint.iacr.org/2022/214.pdf>

TLS SIGNATURES

- “CertificateVerify: A signature over the entire handshake using the private key corresponding to the public key in the Certificate message.” - RFC8446
 - Public key in the end-entity Certificate
 - Certificate is the whole Certificate Chain

TLS 1.3 AND THE PKI

- The signatures could be:
 - two signatures in the certificate chain
 - **one handshake signature**
 - one Online Certificate Status Protocol (OCSP) staple for revocation state
 - two SCTs used for certificate transparency

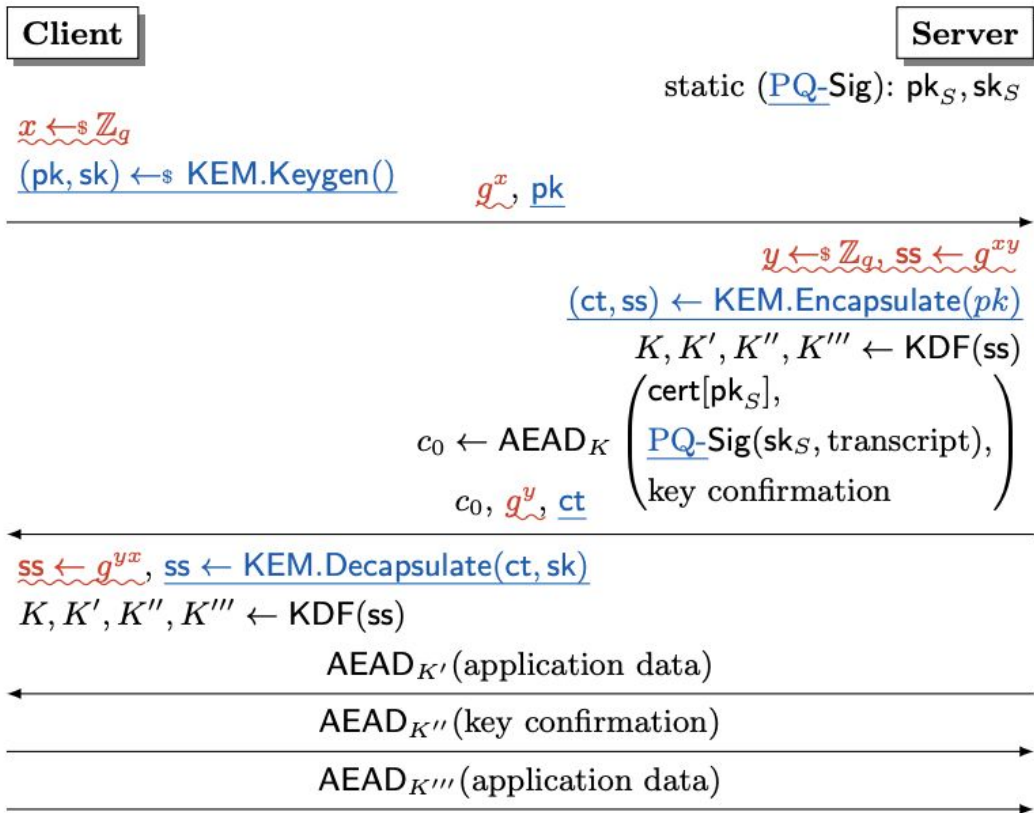


Fig. 1. Comparison between TLS 1.3 handshake (black and red) and PQTLS (black and blue). Note that both protocols require the same number of rounds and messages.

TLS 1.3 HANDSHAKE SIGNATURE

- How much will adding PQ signatures slow connections?
 - Initial study by Westerbaan:
<https://blog.cloudflare.com/sizing-up-post-quantum-signatures/>
 - Missing some verifications and edge cases
- “Online” signature: created every time a TLS connection is made
- If the computational times of PQ signatures is slower and their sizes bigger, can we use KEMs for authentication?
 - Enter KEMTLS or AuthKEM

KEM-BASED AUTHENTICATION

- a.k.a KEMTLS or AuthKEM: abstraction of Diffie–Hellman key exchange
- DH-based authentication is not new: Signal, OTR, Wireguard, OPTLS (first version of TLS 1.3)
- Interactive but not publicly verifiable
- Smaller trusted code base and usage of only one algorithm
- Downside: in TLS 1.3, it increases the number of rounds depending on which peer authenticates:
 - No extra round trips required until client starts sending application data for server-only authentication (half-round trip added)
 - Full round-trip added for mutual authentication

KEMTLS

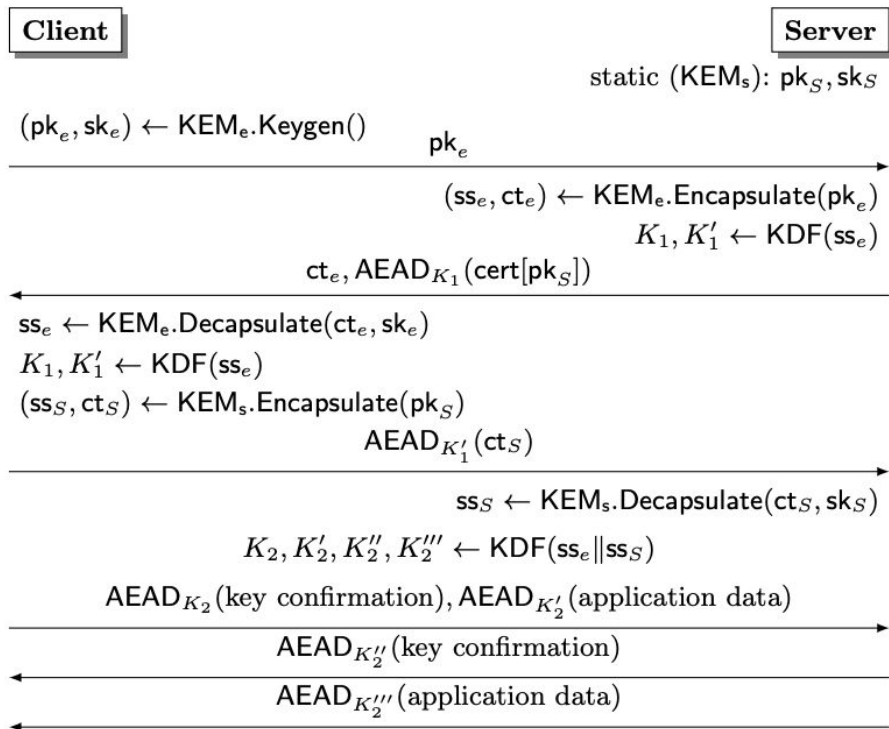


Fig. 2. Overview of KEMTLS with server-only authentication.

KEMTLS

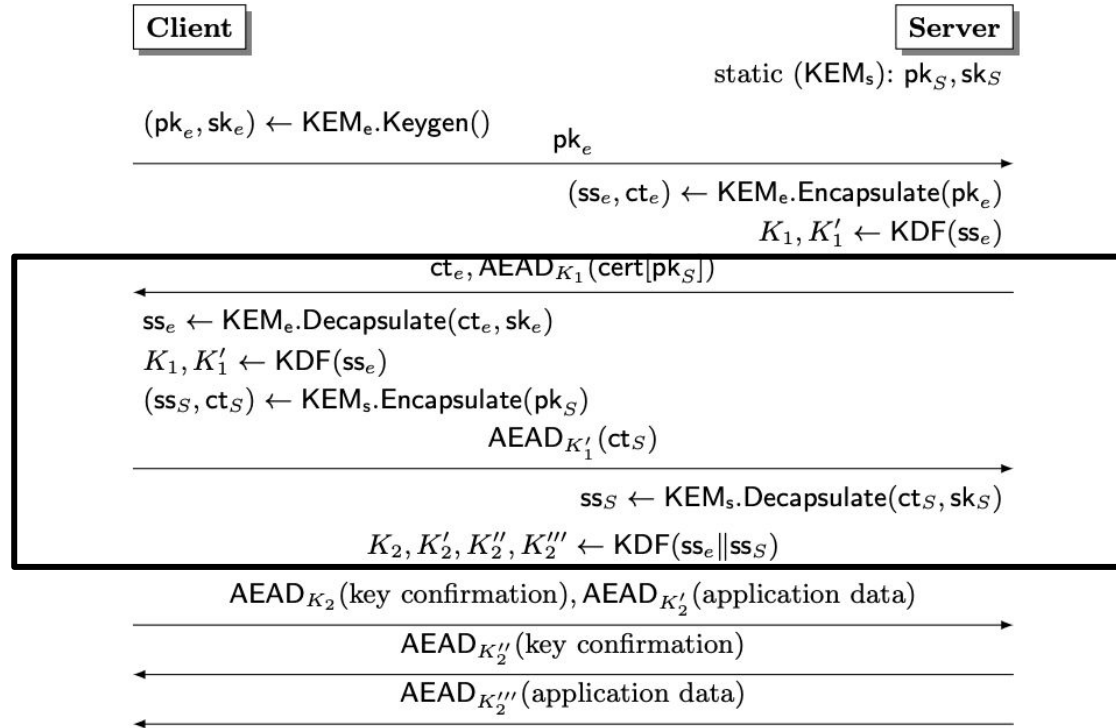


Fig. 2. Overview of KEMTLS with server-only authentication.

- Peter Schwabe, Douglas Stebila, Thom Wiggers
- ACM CCS 2020: <https://eprint.iacr.org/2020/534>
- ESORICS 2021: <https://eprint.iacr.org/2021/779>
- Sofía Celi, Peter Schwabe, Douglas Stebila, Nick Sullivan, Thom Wiggers:
<https://datatracker.ietf.org/doc/html/draft-celi-wiggers-tls-authkem-01>
- Measuring and Implementing KEMTLS:
<https://eprint.iacr.org/2021/1019.pdf>

- Comparison:

Table 3. Average time in 10^{-3} seconds of messages for mutual authentication. Note that timings are measured per-client and per-server: each one has its own timer. The 'KEX' label refers to the Key Exchange and the 'Auth' label refers to authentication.

Handshake	KEX	Auth	Handshake Flight							
			1 st	2 nd	3 rd	4 th	5 th	6 th		
TLS 1.3	X25519	Ed25519	0.113	0.420	111.358	121.349				
TLS 1.3+DC	X25519	Ed25519	0.148	0.546	129.638	178.90				
TLS 1.3+DC	X25519	Ed448	0.154	0.221	137.131	192.283				
PQTLS	Kyber512	Dilithium3	0.125	1.326	231.232	191.187				
PQTLS	SIKEp434	Dilithium4	3.324	7.294	459.888	216.077				
KEMTLS	Kyber512	Kyber512	0.244	0.303	231.752	175.490	375.202	346.308		
KEMTLS	SIKEp434	SIKEp434	2.450	6.206	431.445	228.414	510.591	436.301		

Variants:

- Number of round trips can be decreased if key is cached/pinned/embedded or out-of-band (using DNS)
 - 1 RTT
 - Bandwidth save

Security/Privacy properties:

- All the properties of TLS 1.3:
 - Key indistinguishability
 - Achieves explicit authentication at different points (retroactive explicit authentication)
 - Achieves downgrade resilience at different points
 - Forward secrecy
- **Privacy:** can KEMTLS-PDK be used with ECH?
- **Deniability:** a form of offline deniability is achieved
- Properties verified by pen-and-proof and formal verification using the Tamarin prover (to appear)
 - Modelled following the TLS 1.3 Tamarin model
 - Modelled following a multi-stage key exchange model

Comments:

- What is more important?
 - Round-trips
 - TLS 1.2 and two-round-trip overhead: caused latency
 - Longer handshake messages
 - Longer computations times

THE PKI

- **Certificate Chain:** root Certificate signed by the Certificate Authority (CA), n number of intermediate certificates, and an end-entity (leaf) certificate
- Each certificate in the chain is signed by the entity identified by the next certificate in the chain (this could be malformed as intermediate deprecated certificates can be present, or misordered)
- “Offline” signing: it does not happen at the handshake phase
- Checks the signature of end-entity certificate, and keeps checking certificates until the root certificate, stored in a trust store (OS or browser)

THE PKI

- Intermediate certificates:
 - Have to be included in the certificate chain
 - They can be reduced: can be cached:
<https://datatracker.ietf.org/doc/html/rfc7924>
 - Proposal to use it more for the post-quantum case

THE PKI

- Certificate Transparency:
 - Issued certificates are placed in public logs
 - Issue a signed certificate timestamp (SCT): a promise to include the certificate in the log within some delay
 - SCTs are presented with the certificate to the client: check that the SCTs are valid and other policies. Check inclusion in the log
 - Privacy violation
 - Designated auditors and monitors check that logs are append-only, globally consistent and that logs do not contain any inconsistencies or misissued certificates

THE PKI

- Certificate Revocation:
 - Online Certificate Status Protocol (OCSP):
 - Privacy violation when checking with the log directly
 - OCSP Stapling: the holder of the certificate staples the check
 - CRL and CRLite
 - Huge databases

THE PQ PKI

- Many problems:
 - Too many signatures to verify
 - Few signatures to create
 - Many public keys to transmit
 - Several privacy violations
 - Two databases to append to, and to check to

THE FUTURE OF THE PKI: PQ AND PRIVATE

- Upcoming work:
 - List the challenges of migrating the PKI to PQ
 - Propose solutions:
 - VOPRFs or PIR to prevent privacy violations (some are not PQ)
 - Shorten chains -> proofs of chains
 - Use Cinderella-like certificates:
<https://ieeexplore.ieee.org/document/7546505>

THE FUTURE OF THE PKI: PQ AND PRIVATE

TLS is complex:

- No real studies on migrating the PKI to PQ
- Middle boxes in the way will be slow to update
 - Chrome stopped their experimentation due to this
 - It will inevitably cause latency, especially on the 95% tails.
 - Challenges on the corner-cases: tails or process-intensive operations
- What about QUIC?
- What about DTLS?
- What about password-based authentication?
- User studies:
 - How slow is 'too slow' for users?

THE FUTURE OF THE PKI: PQ AND PRIVATE

“Instead of looking at substituting algorithms, we can look at the properties and how to make them happen in a post-quantum way.”

THE FUTURE OF DNSSEC

- The scenario is DNSSEC is more dim:
 - Avoid fragmentation at all costs
 - Avoid using TCP
 - No load on verification
 - No Round-3 finalist public key fits in a DNS datagram

THE FUTURE OF DNSSEC

- Rethink DNSSEC:
 - What operational challenges are there?
 - How much do DNS attacks occur in the wild?
 - Ongoing work

THE FUTURE

- The quantum world: VOPRFs are usually based in Diffie-Hellman (DDH) or RSA:
 - Isogeny-based proposal: *Oblivious Pseudorandom Functions from Isogenies* by Dan Boneh and Dmitry Kogan and Katharine Woo
 - Lattice-based proposal: *Round-optimal Verifiable Oblivious*
 - *Pseudorandom Functions from Ideal Lattices* by Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart
- Quantum annoyance?: *The “quantum annoying” property of password-authenticated key exchange protocols* by Edward Eaton and Douglas Stebila:

“a scheme is said to be “quantum-annoying” if a quantum computer can compromise the security of the scheme, but only by solving one discrete logarithm for each guess of a password (sic, for example). Considering that early quantum computers will likely take quite long to solve even a single discrete logarithm, a quantum-annoying PAKE, combined with a large password space, could delay the need for a post-quantum replacement by years, or even decades”

THE FUTURE

- Find more of these open questions:
<https://github.com/claucece/HACS-2022-Celi/blob/main/migration-PQ/PQ-Migration.pdf>
- Join the slack for discussion!
https://join.slack.com/t/post-quantumfuture/shared_invite/zt-16gd8po13-~i0ReTJKn3J_F6AZXIJSBA

THANK YOU!

@claucece
