January, 2022

PQC activities in IETF and ETSI CYBER

Kris Kwiatkowski
Senior Cryptography Engineer
PQShield, Oxford/London, UK

PQShield, (UK, NL, FR, USA)

PQShield is helping define the new standards
and developing Quantum-Safe Cryptography Solutions
for Software, Hardware, and Communications

Hardware

A post-quantum System on Chip crypto co-processor toolkit with full emulation

Software

A post-quantum software library integrated with secure comm stacks (TLS, VPN, PKI, E2E).

Research

A high density of PhDs focuses on next generation of cryptographic systems.

Motivation: PQ - threats today

- Complexity
  - **non-trivial complexity**: complicated mathematical background needs to be well studied by the research community before deployment in the industrial environment
  - **Secure implementations takes time**: cryptographic schemes (i.e. side-channel resistance, masking, optimizations), backward compatibility, migration plans

- Lack of standards for communication protocols
  - **Integration** into protocols (like TLS, IKEv2, X509, SSH,...) is just next step after NIST PQC is finalized
  - **Composition** of PQ and non-PQ algorithms
  - **Migration** path supporting backward compatibility
  - Industry requires certain level of **performance**

- “Harvest now, Decrypt latter” attacks.
  - Public key - based encryption can be broken and **decrypted retrospectively**
  - Requires a data center offering a lot of space

Intelligence Community Comprehensive National Cybersecurity Initiative Data Center is a data storage facility designed to store data estimated to be on the order of **exabytes or larger.**
The National Security Agency (NSA) leads operations.

https://en.wikipedia.org/wiki/Utah_Data_Center

Protocols integration

- This talk
  - IETF & ETSI view on PQ TLS (HTTPS), IKE (VPN)
  - Ongoing effort on PKI
- Goals
  - No regression in security
    - Security guarantees of cryptographic schemes
    - Operability
  - Clear migration path
  - Backward compatibility

Protocols integration

- This talk
  - IETF & ETSI view on PQ TLS (HTTPS), IKE (VPN)
  - Ongoing effort on PKI
- Goals
  - No regression
    - Security guarantees of cryptographic schemes
    - Operability
  - Clear migration path
  - Backward compatibility

Protocols integration

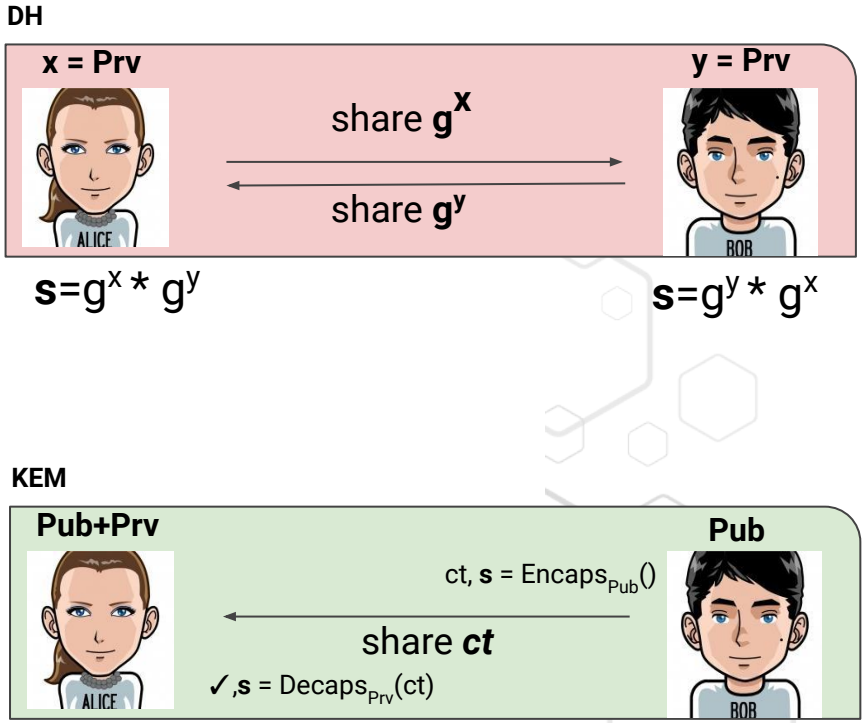
- Internet Engineering Task Force (**IETF**):
  - Main work around PQC is scattered in 3 different groups
    - `tls`: based on 1.3 version (RFC 8446) of the protocol
    - `ipsec`: IKEv2 based VPN
    - `lamps`: X509 and PKI
  - Maybe new workgroup focusing on PQC to be formed
  - In parallel, `cfrg` and `lamps` groups work on hash-based signatures and theirs applications XMSS (RFC8391) and HSS/LMS (RFC 8554, 8708).
- European Telecommunication Standards Institute (**ETSI**):
  - “CYBER QSC” - dedicated workgroup focusing on topics around PQ cryptography
- While waiting for results of NIST PQC, most of the work is focused around **hybrid/composed schemes** and integration into protocols

NIST Round 3 PQ Algorithms

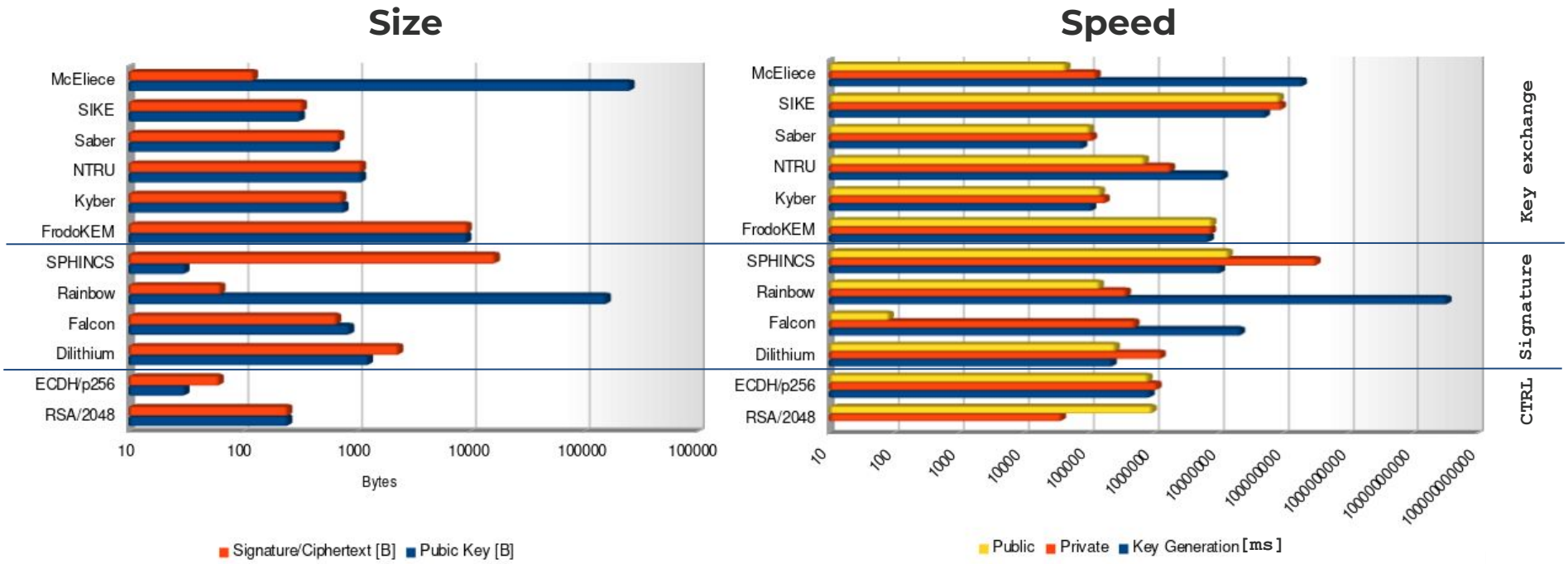
- Public key cryptography
- Key establishment
  - KEM (Key Encapsulation Mechanism) instead of DH
- Signature
  - Almost none of the algorithms use hash-and-sign approach

Diffie-Hellman vs KEM - differences

- KEM Interface
  - Asymmetric: Both sides perform different operation
  - Doesn't fit into DH or PKE interfaces
- Operations
  - Randomized encapsulation
  - Deterministic decapsulation requires both public and private keys
- IND-CCA2 security
  - Shared secret **s** always indistinguishable from random (even if attacker has an ability to decapsulate arbitrary ciphertexts).
  - Security against active attacker



NIST Round 3 PQ Algorithms

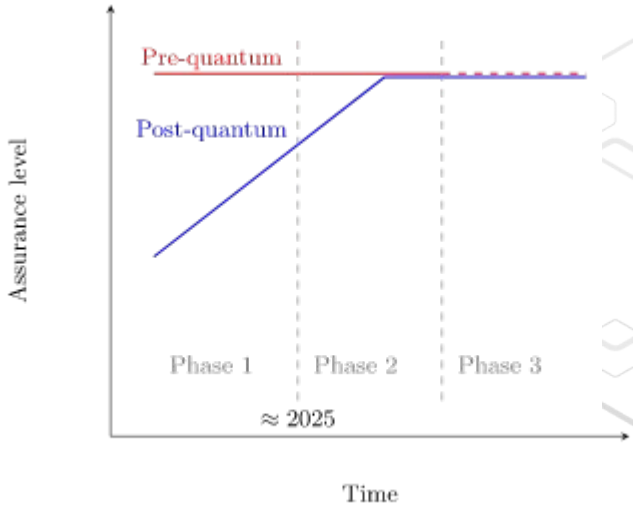


Migration

Multiple players in the industry has already expressed an interest in multistep migration into post-quantum cryptography and using hybrid schemes (i.e. PQC session during ICMC2021).

ANSSI also proposes 3-step process:

- **Phase** 1 (~2022): hybridization to provide some additional post-quantum *defense-in-depth* to the pre-quantum security assurance.
- **Phase 2** (~2025): hybridization to provide *post-quantum security assurance* while avoiding any pre-quantum security regression.
- **Phase 3** (~2030): optional standalone post-quantum cryptography.



Personal opinion

- Preliminary step: migration to the newest version of the protocols
- Key exchange is lower-hanging fruit, goes first

Quantum-Safe TLS

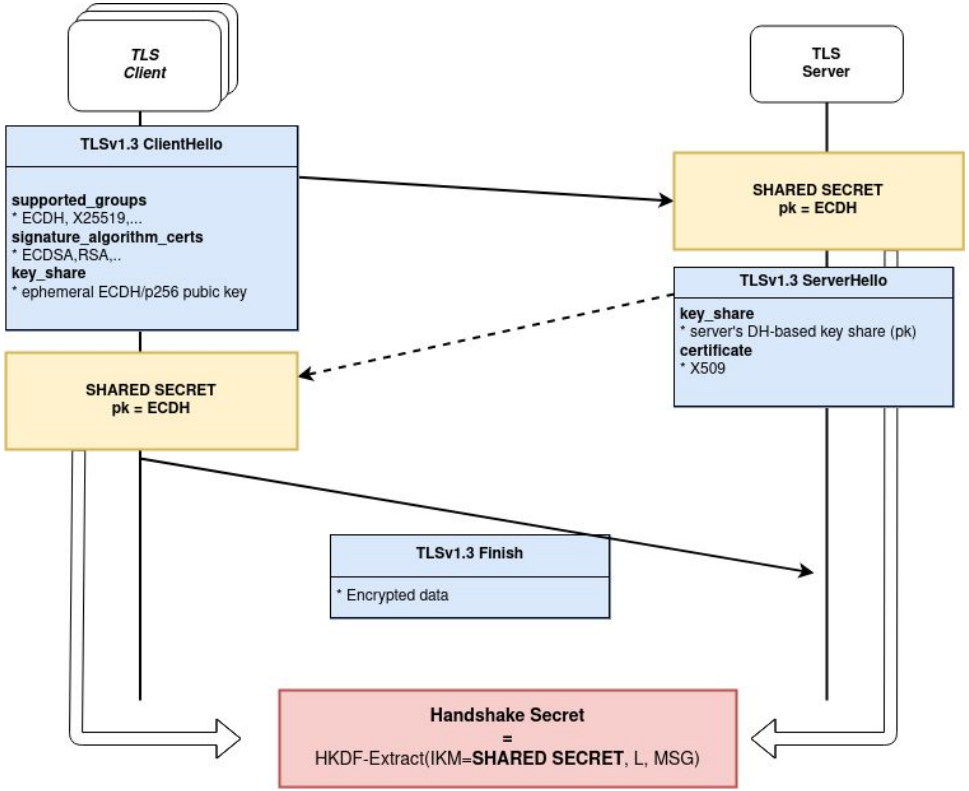
- **Challenges**
  - Interoperability
    - Solution must work with PQ-aware and non-aware clients
    - Middleboxes: It's not just about server & client
  - High performance
    - Solution must be considerate of CPU usage on the server side
  - Low latency
    - Ideally, no extra round trips
    - Solution needs to take into account standard TCP settings
      - Maximum Segment Size (<1500B)
    - How usage of PQ schemes affects TTFSB?
      - *"In e-commerce 1 second delay can result in a 7% reduction in conversions"* (sales) (https://www.bluecorona.com/blog/how-fast-should-website-be/)

TLS v1.3 Session key establishment

supported_groups:
IDs of supported key exchange algorithms

signature_algorithm_certs:
supported signature algorithms

key_share:
default, ephemeral public key



certificate:
certificate signed by one of the algorithms from **supported_algorithm_certs**

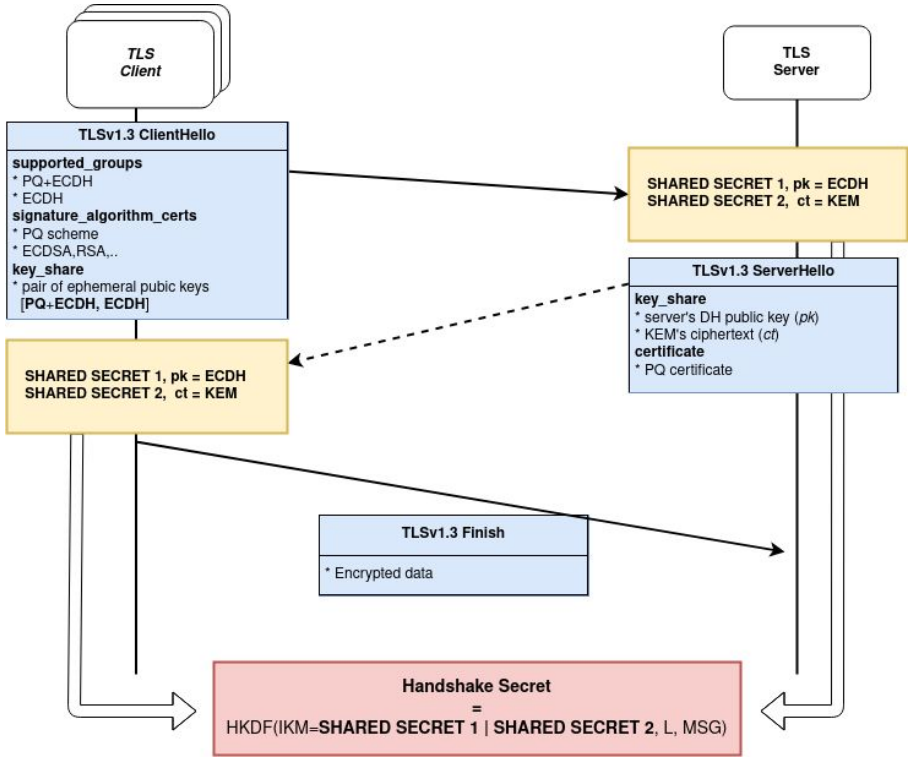
TLS v1.3 Session key establishment - hybrid PQ

supported_groups:
separated ID for each hybrid-PQ "scheme"

signature_algorithm_certs:
separated ID for each hybrid-PQ "scheme"

key_share:
two public keys - classical DH (backward compatibility), and DH+PQ keys concatenated

Client perform KEM key generation and decapsulation.



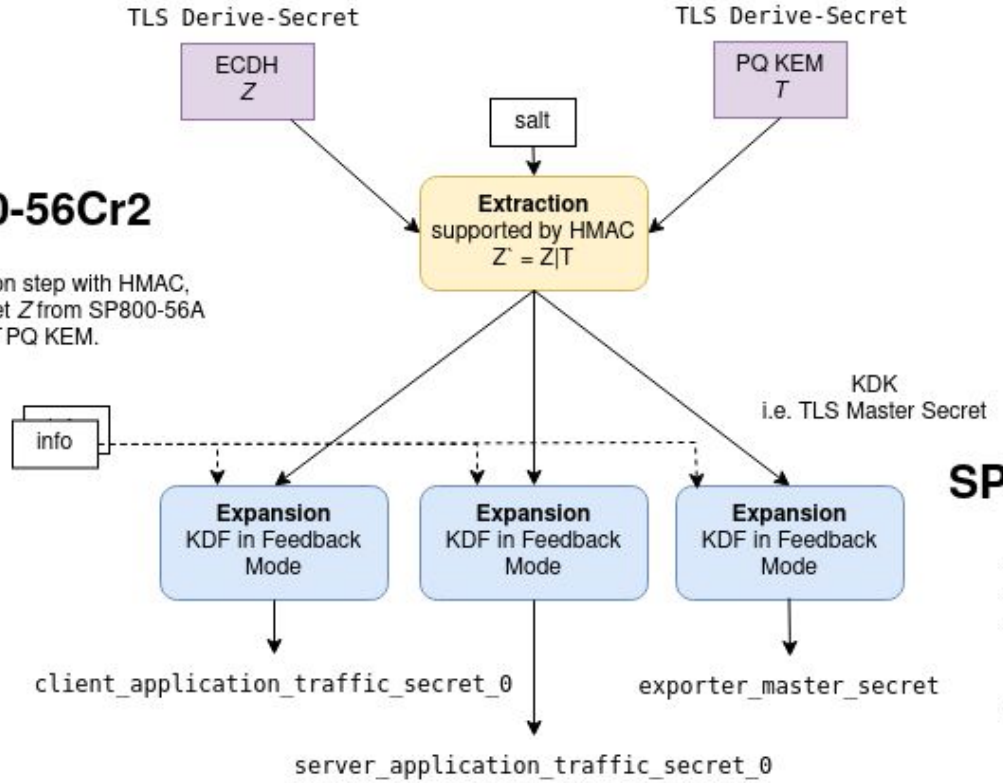
key_share:
Depending on PQ support, server chooses one of the keys

Server performs encapsulation.

TLS v1.3 Session key establishment

- Hybrid PQ key exchange schemes
  - One ID per each combination of PQ+classical schemes
  - Concatenation of public keys and shared secrets (no structure)
  - Backward compatibility
    - Client sends min. 2 key shares - hybrid and classical
    - **Pros**: simplicity, **Cons**: duplication of data
  - Forward compatibility
    - TLS HelloRetryRequest used in case different PQ scheme supported by the server (useful during migration)
  - IETF Drafts: (first) `draft-kiefer-tls-ecdhe-sidh-00`, (then) `draft-ietf-tls-hybrid-design-04`
- FIPS 140-3 compliant TLS key schedule
  - Compliance with NIST TLS 1.3 KDF (SP 800-133 r2, SP 800-108 and SP 800-56C r2)

FIPS compliance rationale



SP800-108

- KDF in Feedback Mode
- HMAC as PRF
- The KDK from extraction step followed by multi-step expansion (see 56Cr2, 5.3)
- 8-bit counter, "0" IV

Extract

&

Expand

Security

- Analytical standpoint
  - Concatenation of two keys is modeled as dual-PRF
    - dual-PRF is a PRF when either of both keys guarantees pseudo-randomeness of the output, even if one of the keys is maliciously chosen (or broken) (Bellare and Lysyanskaya *"Symmetric and Dual PRFs from Standard Assumptions: A Generic Validation of an HMAC Assumption"*, 2015)
    - HKDF-extract (based on SHA2) can be modeled as dual-PRF combiner (Bindel et al. *"Hybrid key encapsulation mechanisms and authenticated key exchange."*, 2019)
  - Shared secrets used by HKDF-extract have fixed length
  - TLS v1.3 permits to use SHA2-256 and SHA2-384 in the key schedule. Both are believed to be quantum-safe (against Grover)
- FIPS
  - No security declared on PQ generated shared secret (can't declared as CSP nor as PSP)
  - Concatenation of secret keys was already allowed by SP800-133r1 (see point 6.6)

Experimenting with the PQ TLS

- Google: CECPQ1 runs *NewHope* scheme in TLSv1.2 (2016)
  - No major problems reported; latency increased by 20ms for 5% of the slowest connections
- Google: NIST candidates (2018)
  - Adam Langley to measure impact of key size on latency. Implemented “dummy” extension to simulate larger key sizes.

    Control group: no extension sent
    Supersingular isogenies (**SI**): 400 bytes
    Structured lattices (**SL**): 1 100 bytes
    Unstructured lattice standing (**ULS**): 3 300 bytes

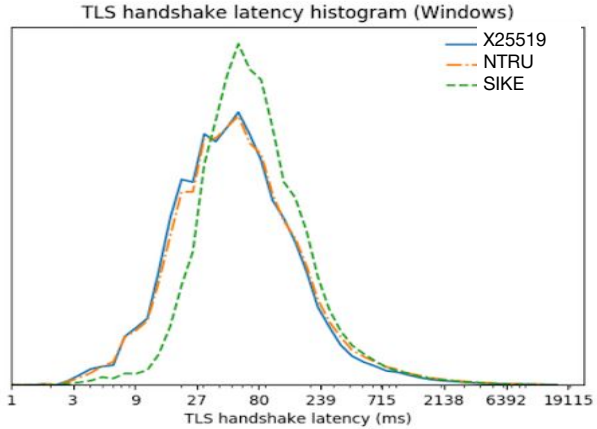
    USL: impractical for TLS
    SL : no major issues found
    SI : maybe too slow for most of the connections but preferable for the slowest 5%
- Google & Cloudflare: CECPQ2 (2019)
  - NTRU-HRSS and SIKE released in the Google Chrome (Canary) and deployed on production servers of Cloudflare CDN & Google Cloud
  - Goal: look for differences in latency, CPU utilization and “Unknown-Unknowns” in real-world scenario
  - Impact of TCP Maximum Segment Size

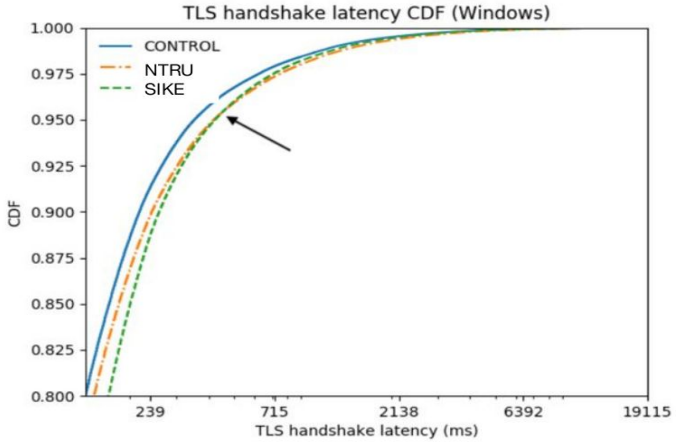
CECPQ2 (Kwiatkowski, Lagnley, Sullivan, Valenta, 2019)

	X25519 (ECC)	NTRU (lattice)	SIKE (isogeny)
PK size	64	1138	~340
Keygen (op/sec)	6049	902	242
Encaps/KEX (op/sec)	2374	25877	148
Decaps/KEX (op/sec)	2374	6312	138

Intel E3-1220v3 (haswell)

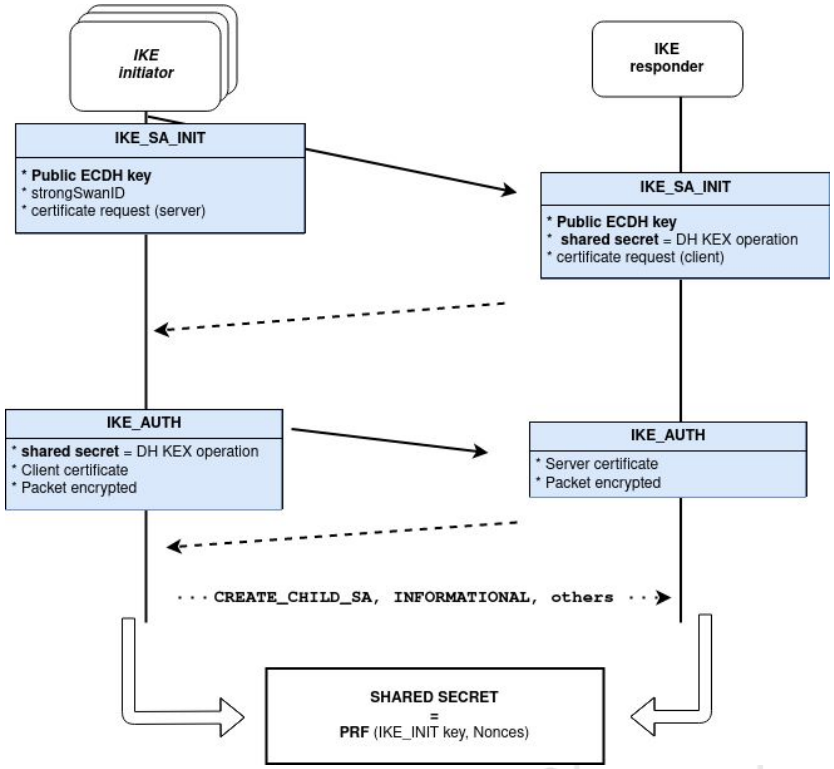
	X25519 (ECC)	NTRU+X25519 (lattice)	SIKE+X25519 (isogeny)
Mobile	[133,135]	[138, 140]	[196, 198]
Desktop	[55, 55]	[58, 58]	[74, 76]





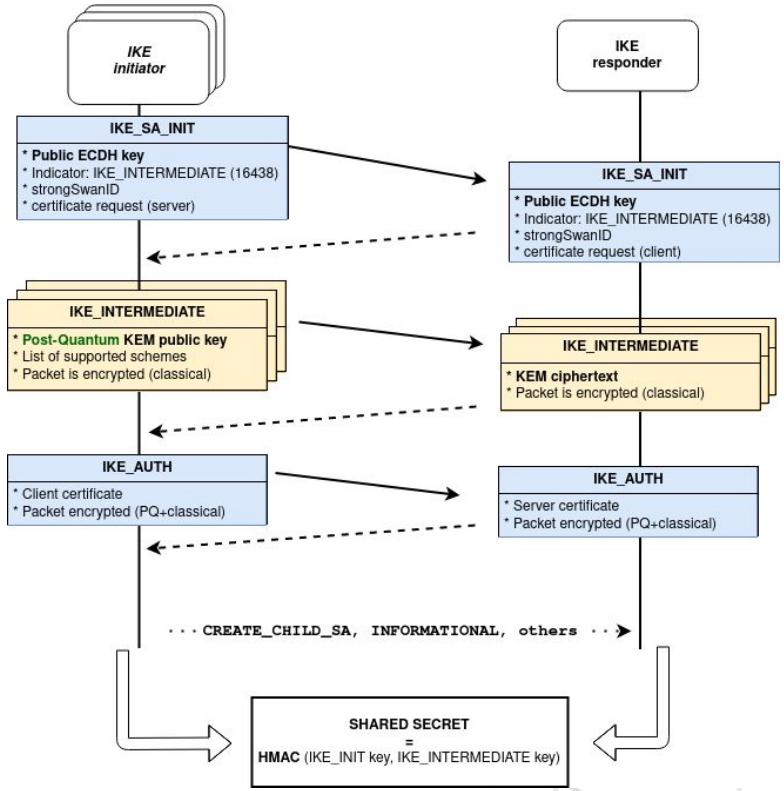
VPN & IKEv2 Protocol

- **Challenge:** a large PQ public keys may cause IKEv2 failures
  - Firewalls may drop larger, fragmented IKE_SA_INIT
    (MTU often set to ~1500 bytes)
- **Protocol:**
  - Request/response message pairs used in communication over UDP (rarely TCP)
  - DH public key sent in the first message exchange, called IKE_SA_INIT
  - There are multiple shared secretes used in communication. All are derived from DH-shared secret established during IKE_SA_INIT.
  - Master shared secret calculated after IKE_SA_INIT is finished
  - Second exchange, IKE_AUTH, used for authentication



Hybrid, Quantum-Safe, IKEv2 Protocol

- **Challenge:** a large PQ public keys may cause IKEv2 failures
  - Firewalls may drop larger, fragmented IKE_SA_INIT
    (MTU often set to ~1500 bytes)
- **Solution:** modify IKEv2
  - IKEv2 support fragmentation but only after IKE_SA_INIT is exchanged (RFC 7383)
  - Additional message (IKE_INTERMEDIATE) to carry large data packets exchanged after IKE_SA_INIT
  - Post-quantum KEM keys and ciphertexts sent during second flight.
  - IETF drafts:
    - `draft-ietf-ipsecme-ikev2-intermediate-07`
    - `draft-ietf-ipsecme-ikev2-multiple-ke-04`
    - `RFC 8784 (pre-shared key)`



ETSI : Quantum-safe Hybrid Key Exchanges (TS 103 744)

CatKDF

Form ***secret*** = psk || k_1 || k_2|| …

Set ***f_context*** = f(context, msg_A, msg_B)

session_key = KDF(***secret***,***f_context***, …)

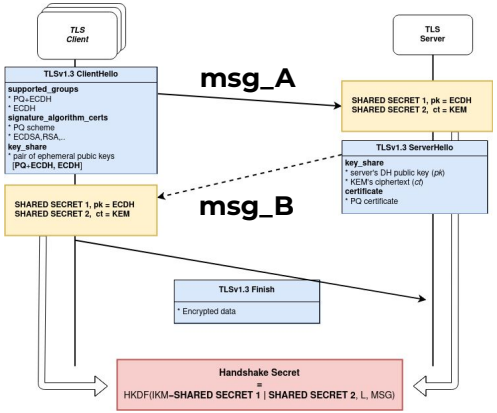
CasKDF

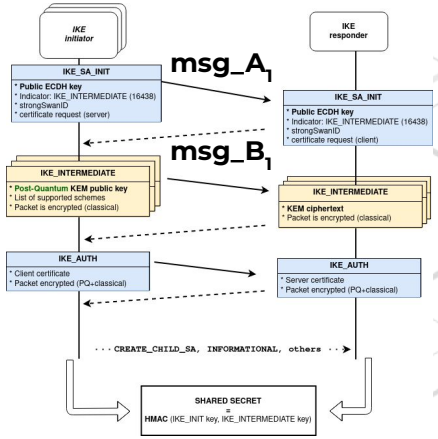
Set $\mathbf{cs_0}$ = psk || digest_len (or 0)

For i = 1 … n:

$\mathbf{rs_i}$ = PRF($\mathbf{cs_{i-1}}$, k_i, msg_A_i, msg_B_i)

cs_i || $session\_key_i$ = KDF($\mathbf{rs_i}$, . . .)





PKI for authentication

Challenges

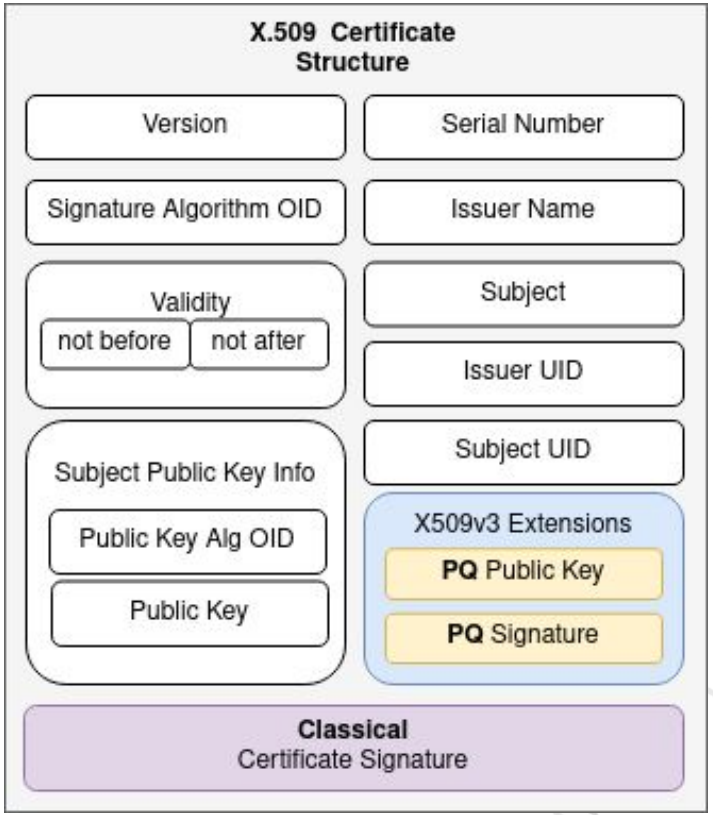
- Unobvious migration path for X509 certificates
- Complicated logic X.509 both for signing and verification
- Less time-critical as quantum-safety is needed only after LSQC is available (not susceptible to “Harvest now, Decrypt latter”)

Solutions discussed in IETF LAMPS group:

- Usage of multiple certificate chains
  - Use cases limited to online, interactive protocols that support negotiation like TLS. One certificate chain uses PQ certificates, the other uses classical cryptography.
  - Doesn’t solve offline, non-interactive use cases (like code-singing)
- X509v3 Extension dedicated to PQ signature
- Composite signatures (aka Dual signatures)

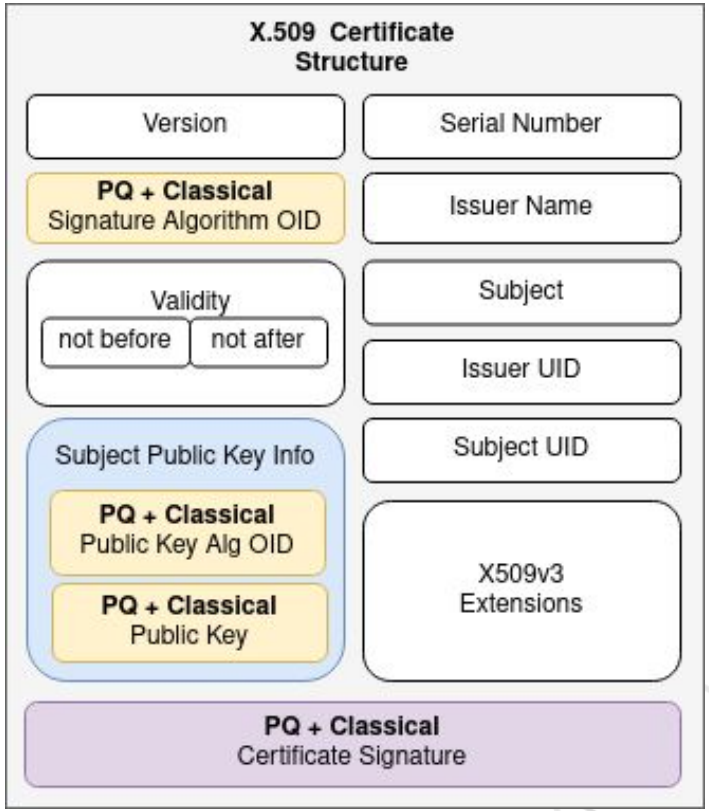
X509: dedicated extensions

- X509v3 extensions allow including additional non-critical fields with arbitrary data
- Additional fields containing a PQ Public Key and a PQ signature.
- PQ signature is verified before or after authenticity of the certificate is checked with classical algorithm and only by already updated systems (backward compatibility)
- Non-critical extension is vulnerable to stripping attacks.
- IETF draft:
  `draft-truskovsky-lamps-pq-hybrid-x509`



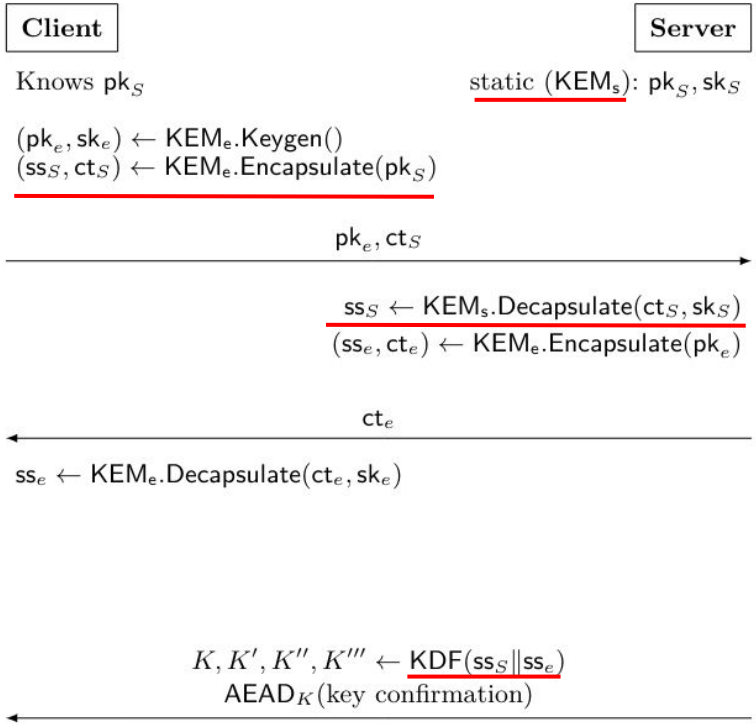
X509: Composite/Dual certificates

- Composition of two - classical and post-quantum schemes
- Certificates signed once with classical and then post-quantum signature scheme
- IETF draft
  - `draft-ounsworth-pq-composite-sigs`
- Composite algorithms identified by OID
  - Single OID identifying *composite* structure with internal identifiers for signature algorithms
  - OID assigned to pre-defined pair of algorithms
- Systems must be updated to be able to composite-certificates is used (no backward compatibility)



Supporting IETF drafts/standards

- KEM for authentication in TLS
  - KEMTLS: use IND-CCA2 secure KEMs for authenticating handshake messages
  - `draft-celi-wiggers-tls-authkem`
- PKI for encryption & CMS
  - `draft-ounsworth-pq-composite-encryption`
  - `draft-perret-prat-lamps-cms-pq-kem`
- Compression of public keys and signatures
  - `draft-ounsworth-pq-external-pubkeys`
- TLS subcerts
  - `draft-ietf-tls-subcerts-11`
- Scheme for hybrid public key encryption (KEM/KDF/AEAD)
  - `draft-irtf-cfrg-hpke-12`



Questions

Supporting slides

History

1994

Peter Shor

Introduces quantum attack on classical **asymmetric** cryptosystems.

Computation complexity of number factorization problem is reduced to $\mathbf{O(\log N)^3}$

In practice it means all currently deployed cryptosystems can be broken on *large-scale* quantum computer.

1996

Lov Grover

Introduces quantum algorithm which improves searching in the unordered set.

This improves best attacks on **symmetric** cryptosystems (AES/SHA).

The problem can be solved by switching to twice longer secret keys (i.e. AES-128 to AES-256).

2016

NIST standardization process

NIST (National Institute of Standards and Technology) starts multi-year project to select a new **asymmetric** cryptosystems resistant to potential attacks by quantum adversaries. Planned end date is Jan 2022.

"It is desirable to plan for this transition early.", NIST

Raport NISTIR 8105

Resource estimations

Computational strength of quantum computers is measured in *qubits*. It has been estimated that computer with a power of 2^2330 (2330 qubits) are needed to break the most commonly used key exchange scheme used currently on the Internet. The IBM machine is currently the strongest (publically known) quantum computer, it has 127 qubits. The large-scale quantum computers are expected to be publicly available in around 10-15 years.

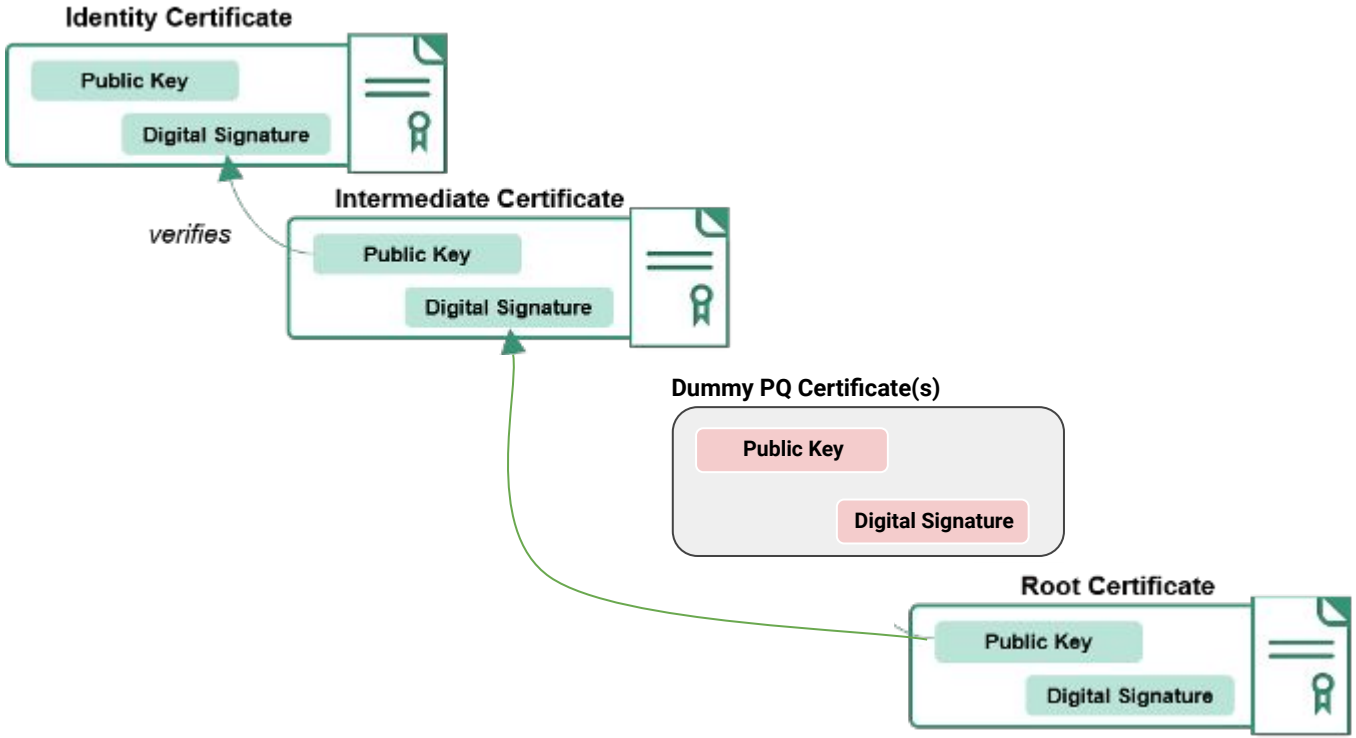
Current availability

- Google: 72 qubits
- IBM: 127 qubits
- Honeywell: 64 qubits

ECDLP in $E(\mathbb{F}_p)$ simulation results					Factoring of RSA modulus N interpolation from [21]		
$\lceil \log_2(p) \rceil$ bits	#Qubits	#Toffoli gates	Toffoli depth	Sim time sec	$\lceil \log_2(N) \rceil$ bits	#Qubits	#Toffoli gates
110	1014	$9.44 \cdot 10^9$	$8.66 \cdot 10^9$	273	512	1026	$6.41 \cdot 10^{10}$
160	1466	$2.97 \cdot 10^{10}$	$2.73 \cdot 10^{10}$	711	1024	2050	$5.81 \cdot 10^{11}$
192	1754	$5.30 \cdot 10^{10}$	$4.86 \cdot 10^{10}$	1 149	–	–	–
224	2042	$8.43 \cdot 10^{10}$	$7.73 \cdot 10^{10}$	1 881	2048	4098	$5.20 \cdot 10^{12}$
256	2330	$1.26 \cdot 10^{11}$	$1.16 \cdot 10^{11}$	3 848	3072	6146	$1.86 \cdot 10^{13}$
384	3484	$4.52 \cdot 10^{11}$	$4.15 \cdot 10^{11}$	17 003	7680	15362	$3.30 \cdot 10^{14}$
521	4719	$1.14 \cdot 10^{12}$	$1.05 \cdot 10^{12}$	42 888	15360	30722	$2.87 \cdot 10^{15}$

"Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms", Roetteler et al. https://arxiv.org/pdf/1706.06752.pdf

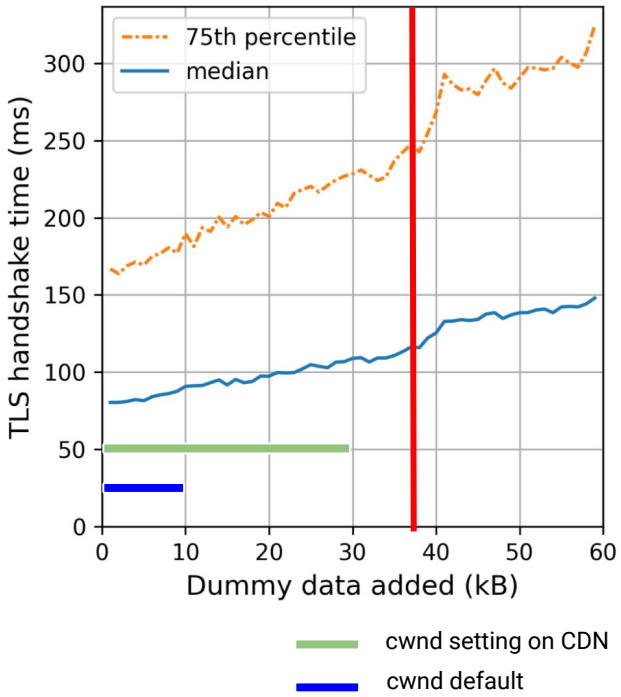
TLS authentication: Cloudflare experiment (2021)



https://blog.cloudflare.com/sizing-up-post-quantum-signatures/

TLS authentication: Cloudflare experiment (2021)

	ECDSA	RSA	Dilithium	Falcon	SPHINCS+
Public key (B)	64	256	1320	897	32
Signature (B)	64	256	2420	668	7856
Signing	1	25	2.5	5	3000
Verification	1	0.2	0.3	0.3	1.7



- 1 TLS session == typically ~6 certificates & public keys

https://blog.cloudflare.com/sizing-up-post-quantum-signatures/

Previous migrations