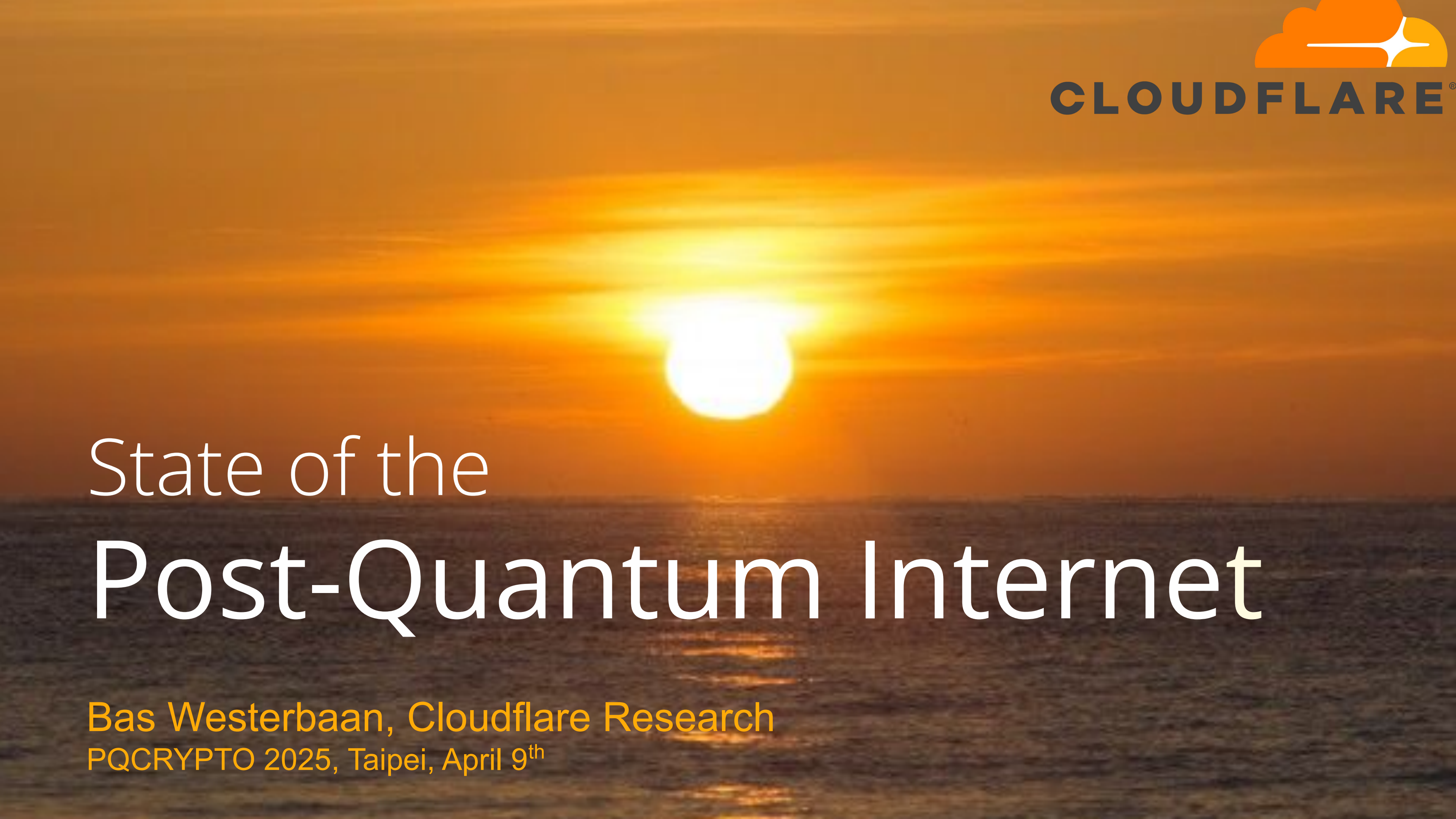


CLOUDFLARE®

State of the Post-Quantum Internet

Bas Westerbaan, Cloudflare Research

PQCRYPTO 2025, Taipei, April 9th

1. Where are we with the migration?
2. What challenges lay ahead.
3. ~~What can we learn going forward.~~
   PQ paradoxes.

But first…

Thank you! 🙏

Post-quantum key agreement is used at a huge scale today.
(Signal, iMessage, webservers, browsers, ...)

About Cloudflare

We run a global network spanning 335 cities in over 120 countries.

Started of as a CDN and DDoS mitigation company, we now offer many more services, including

- 1.1.1.1, public DNS resolver
- Workers, developer platform
- Zero trust, to protect corporate networks

We serve nearly 20% of all websites and process 71 million HTTP requests per second. >35% of Fortune 500 are paying customers.

Building a better Internet

Cloudflare cares deeply about a private, secure and fast Internet, helping design, and adopt, among others:

- Free TLS certificates (2014), TLS 1.3 and QUIC
- DNS-over-HTTPS
- Private Relay / OHTTP
- Encrypted ClientHello

And, you will not be surprised:

- Migrating the web to Post-quantum cryptography.

What we already knew going in

Changing the Internet / WebPKI is hard

- Very diverse. Many different users / stakeholders with varying (performance) constraints and update cycles.

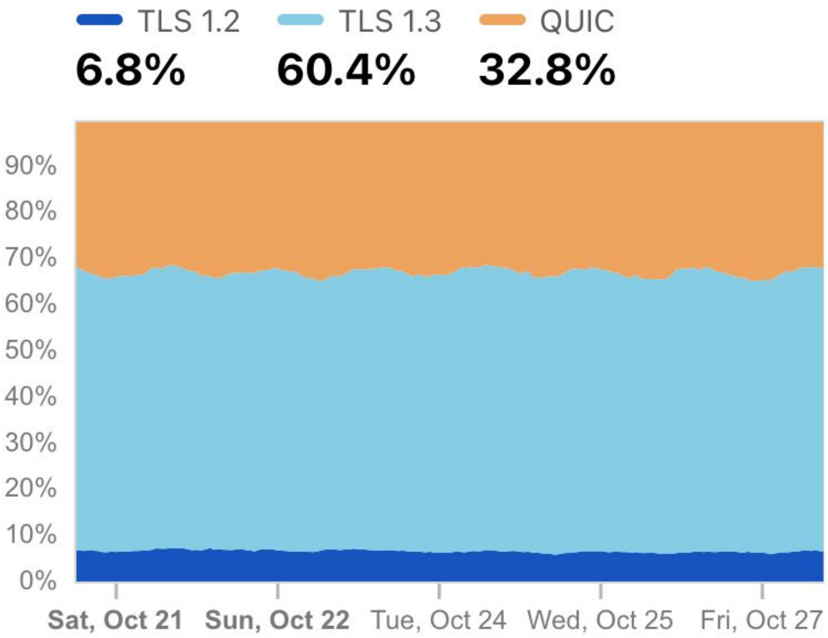
  We can't assume everyone is on fiber, or uses modern CPU, can store state, or can update at all.

- Protocol ossification. Despite being designed to be upgradeable, any flexibility that isn't used in practice is probably broken, because of faulty implementations.

TLS 1.3 migration

Early versions of TLS 1.3 were completely undeployable because of protocol ossification.

After six more years of testing and adding workarounds, the final version of TLS 1.3 is a success, used by over 90% of our visitors.



Cloudflare Radar

1. **Key agreement 🤝**
   Store now; decrypt layer: upgrading is urgent.

2. **Signatures 🖋️**
   Seems less pressing: we need to upgrade and rotate before the arrival of the CRQC.

Key agreement 🤝

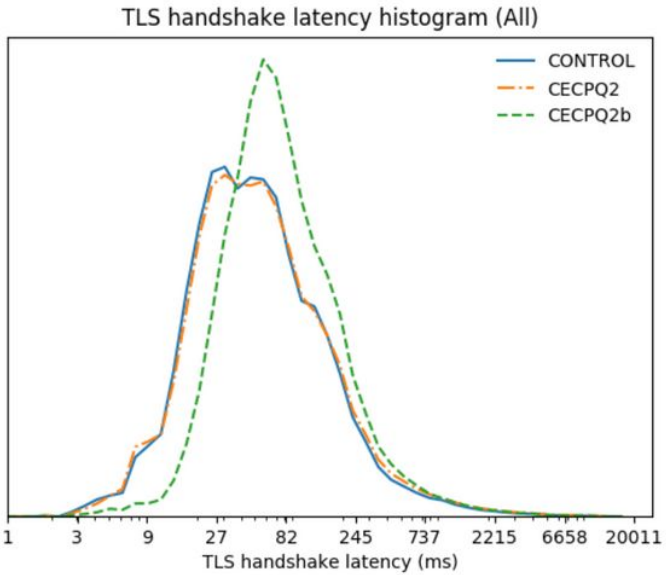
Urgent, and the *easier* one.

Feasibility study with Chrome

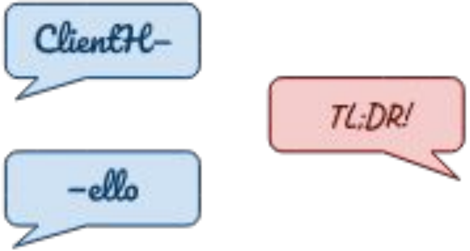
In 2019 we performed large-scale test of PQ kex with Chrome. Takeaways:

- Performance of lattice-based KEMs is acceptable, even as *hybrid*.
- Significant amount of broken clients because of protocol ossification (*split ClientHello*.)

Google has been working with vendors to fix issues.



X25519. CECPQ2 is X25519+NTRU-HRSS (lattice) and CECPQ2b is X25519+SIKE (isogenies, broken)



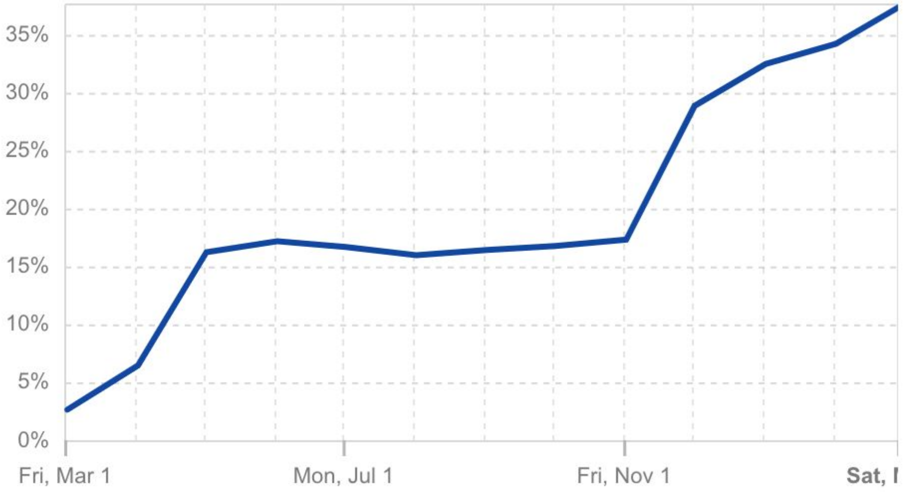
Adoption

2022 coordinating at IETF, we enabled hybrid post-quantum key agreement (~20% Internet.)

In 2023 Google enabled server-side as well.

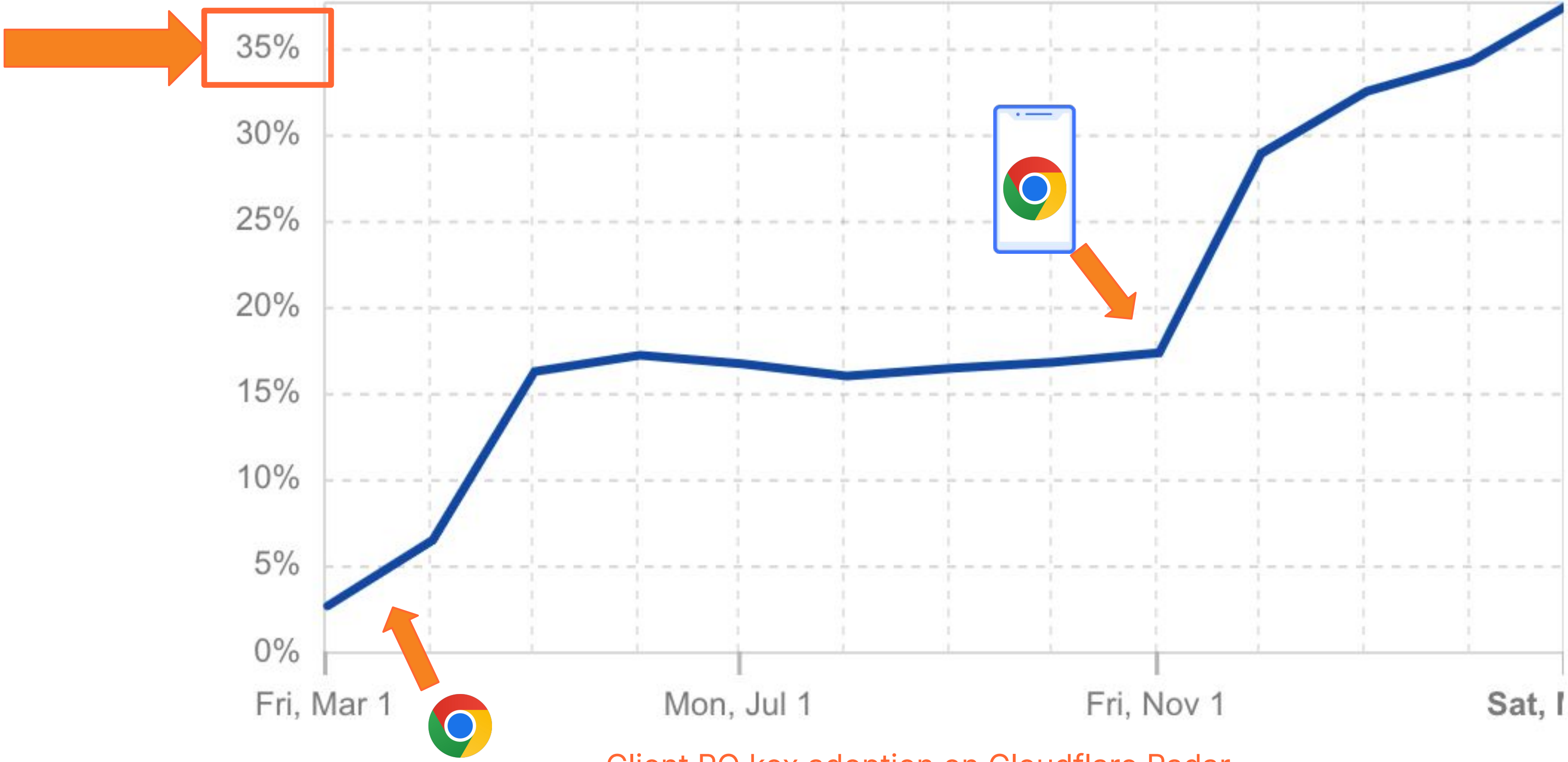
Clients:

- Chrome, Firefox, and Edge: enabled by default.
- Safari: experimenting, we see 1–2% traffic PQC.



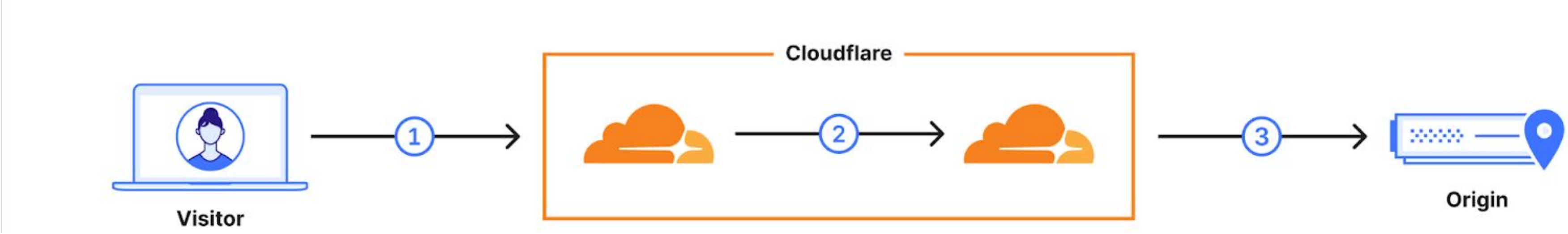
Client PQE adoption on Cloudflare Radar

- Go & Zig enable by default.
- OpenSSL 3.5 (released today!) enables by default.



Client PQ kex adoption on Cloudflare Radar

Post-quantum to origins



We enabled support for PQ key agreement to origins (3).

0.9% of origins support PQ at the time of writing

0.34% incompatible when sending keyshare immediately. We’ve reached out to customers to help remediate.

Long tail of weird bugs (atomic fragments, reordering, load balancers) discussed in Suleman Ahmad’s talk.

Far from just a technical challenge

In 2023 we commenced migrating our internal connections to post-quantum key agreement.

Huge effort: every engineering team created inventory of cryptography used, risks, and planned/executed migration.

Majority of our internal connections are secured (prioritizing sensitive connections), but a long tail remains.

On the upside: we did not encounter any performance or compatibility issues.

Key agreement 🤝

Urgent and the easier of the two to deploy; with ~38% client adoption, the new modern baseline for the Internet.

That took 6 years.

1. Where are we with the migration?
2. What challenges lay ahead.
3. PQ paradoxes.

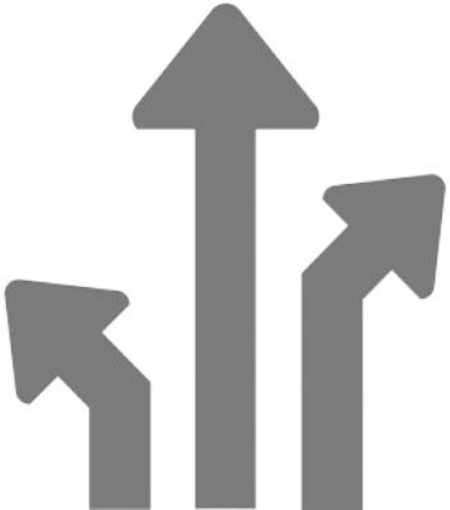
Signatures 🖋️

Less urgent, but much more challenging.

#1, many more parties involved:

Cryptography library developers, browsers, certification authorities, HSM manufacturers, CT logs, and every server admin that cobbled together a PKI script.

Not just software update: also key rotation.



#2, there is no all-round great PQ signature

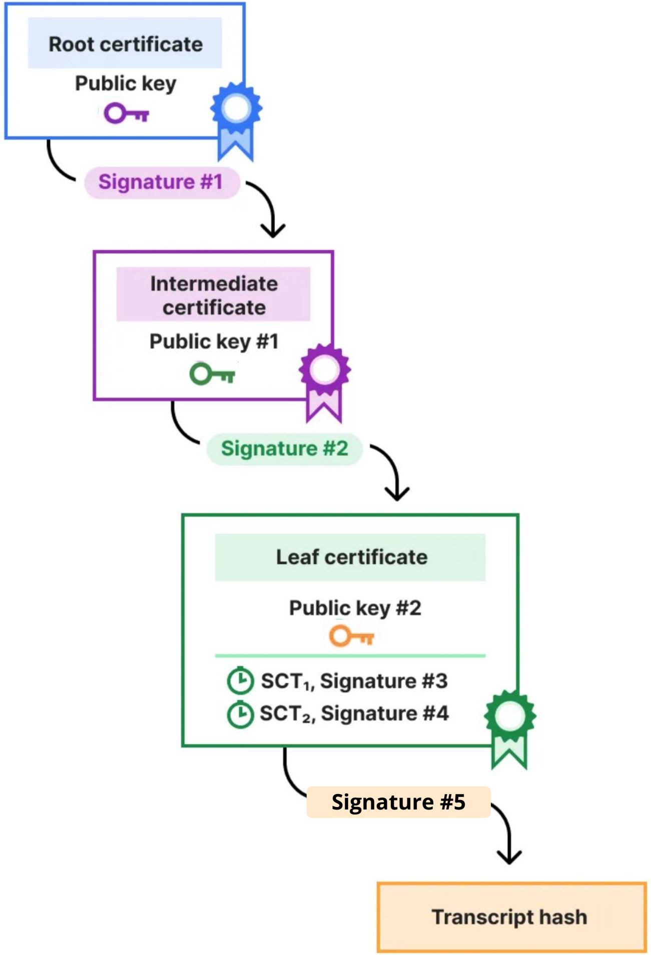
		Sizes (bytes)		CPU time (lower is better)	
	PQ	**Public key**	**Signature**	**Signing**	**Verification**
Ed25519	❌	32	64	0.15	1.3
RSA$_{2048}$	❌	256	256	80	0.4
ML-DSA $_{44}$	✅	1,312	2,420	1 (baseline)	1 (baseline)
SLH-DSA$_{128s}$	✅	32	7,856	14,000	40
SLH-DSA$_{128f}$	✅	32	17,088	720	110
LMS$_{M4\_H20\_W8}$	✅	48	1,112	2.9 ⚠️	8.4
FN-DSA$_{512}$ (Falcon)	✅	897	666	3 ⚠️	0.7

Online signing — FN-DSA’s Achilles’ heel

- For fast signing, FN-DSA requires a floating-point unit (FPU).
- We do not have enough experience running cryptography securely (constant-time) on the FPU.
- On commodity hardware, we’re uncomfortable using FN-DSA when signature creation can be timed, eg. TLS handshake.
- Not a problem for signature verification or offline signatures.

#3, there are many signatures on the Web

Typically 5 signatures and 2 public keys when visiting a website.



Using only ML-DSA-44

+17,144 bytes

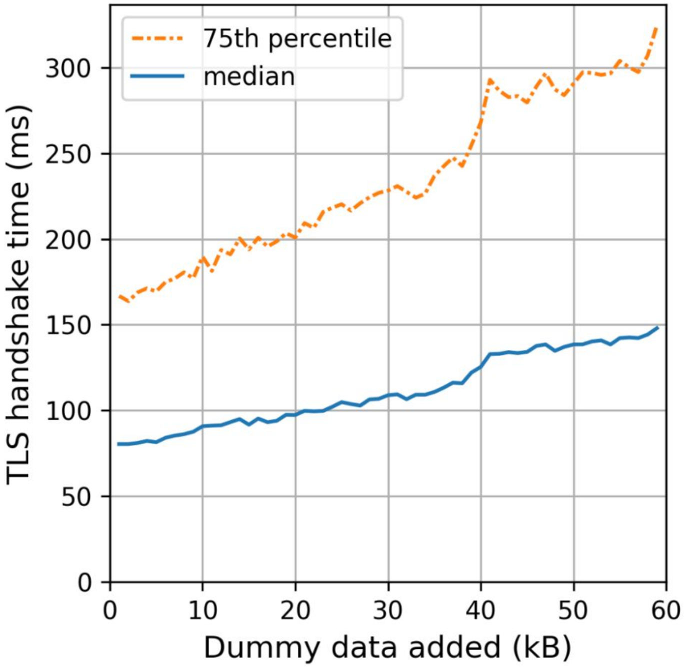
Using ML-DSA for the TLS handshake and FN-DSA for the rest

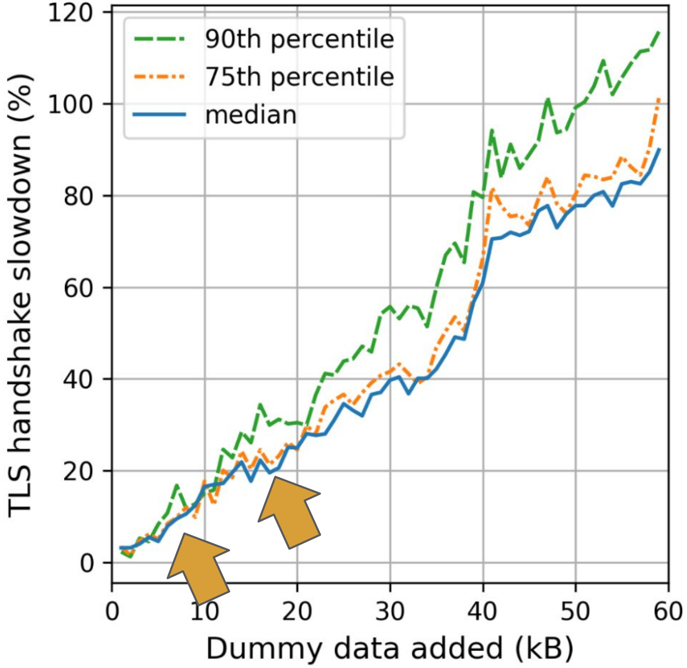
+7,959 bytes

Is that too much? We had a look…

How many (bytes) is too many?

Sizing up post-quantum signatures, 2021: We found that every 1kB added to the handshake slows it down by about 1.5% at the median.





How many (bytes) is too many?

Sizing up post-quantum signatures, 2021: We found that every 1kB added to the handshake slows it down by about 1.5% at the median.

Chromium Security Design Principles, 2024: “Adding ~7kB is implausible unless a cryptographically relevant quantum computer (CRQC) is tangibly imminent.”

How many (bytes) is too many?

Sizing up post-quantum signatures, 2021: We found that every 1kB added to the handshake slows it down by about 1.5% at the median.

Chromium Security Design Principles, 2024: “Adding ~7kB is implausible unless a cryptographically relevant quantum computer (CRQC) is tangibly imminent.”

Another look at PQ signatures, 2024: Median bytes transferred from server to client for the lifetime of non-resumed QUIC connections to Cloudflare is 7.8kB including certificates.

Given the median compressed chain is 3.2kB…

For us today certificates account for more than 40% of data transferred over half the QUIC connection.

How significantly does it affect actual end-user performance? We'd say it doesn't look good, but opinions differ. We'll run another experiment.

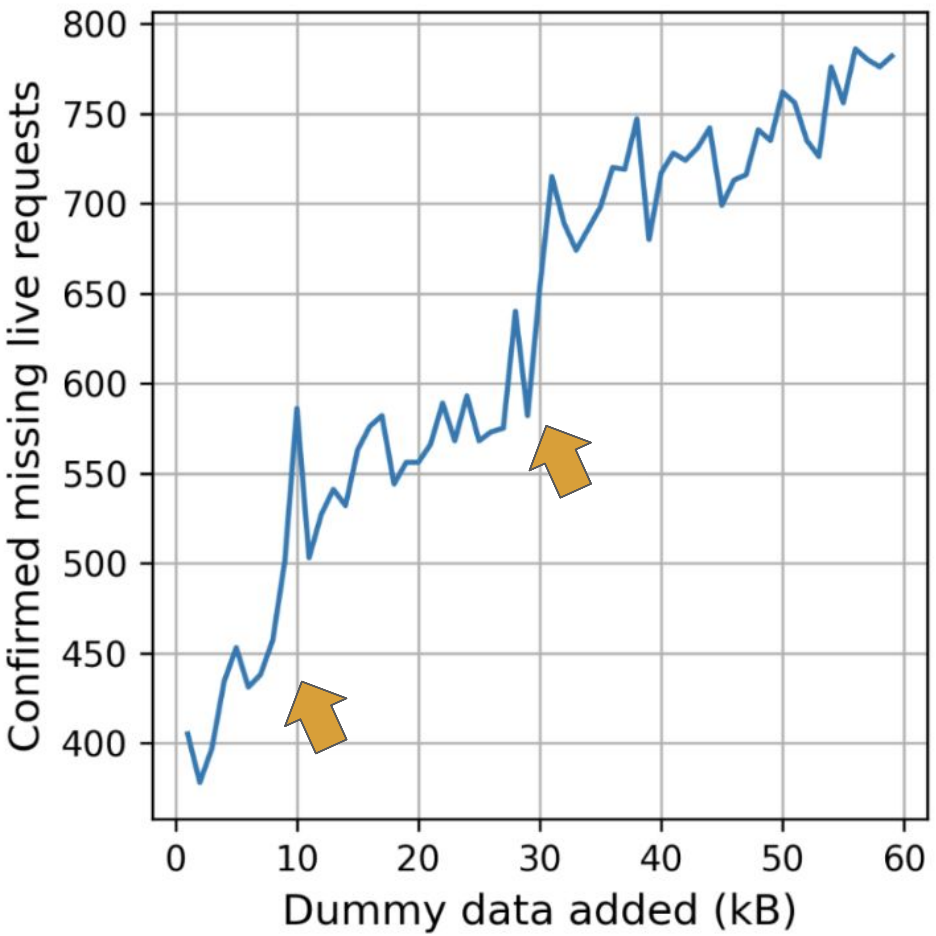
And, of course…

Protocol ossification

Bump in missing requests suggests some clients or middleboxes do not like certificate chains longer than 10kB and 30kB.

This is problematic for single certificate migration.

Instead configure servers with multiple separate certificates and let TLS negotiate the one to send.



Not great, not terrible

It probably won’t break the Web, but the performance impact will delay adoption.

NIST signature on-ramp

As you know, NIST has a new competition for more signature schemes, which is in its second round.

The short of it: there are some very promising submissions, but their security is as of yet unclear.

Thus, we cannot assume that a new post-quantum signature will solve our issues.

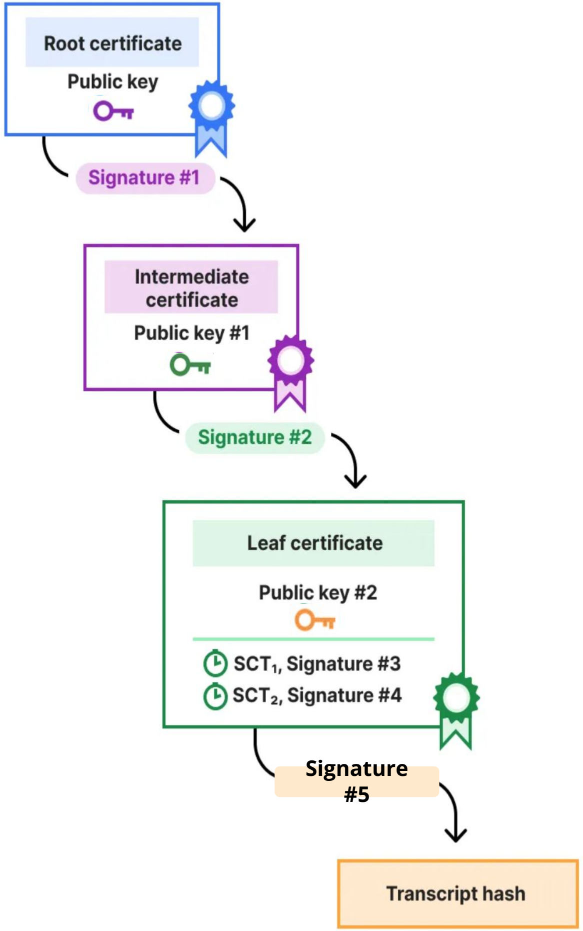
Concrete instances with on-ramp candidates

Using MAYO *one* for leaf/intermediate, and *two* for the rest, adds 3.5kB. Signing time between ECC/RSA.

Using UOV ls-pkc for root and SCTs, and HAWK512 for the rest, adds 3.2kB. 66kB for stored UOV public keys.

Using UOV ls-pkc again, but combined with ML-DSA$_{44}$. Adds 7.4kB. More conservative choice.

SQIsign only. Adds 0.5kB. Signing time >1s (not constant-time), and verification time >35ms. 🐢



In the meantime

There are small and larger changes possible to the protocols to reduce the number of signatures.

- Leave out intermediate certificates.
- Use key agreement for authentication.
- Overhaul WebPKI, eg. Merkle Tree Certificates or KT-inspired Starlit Jellyfish.

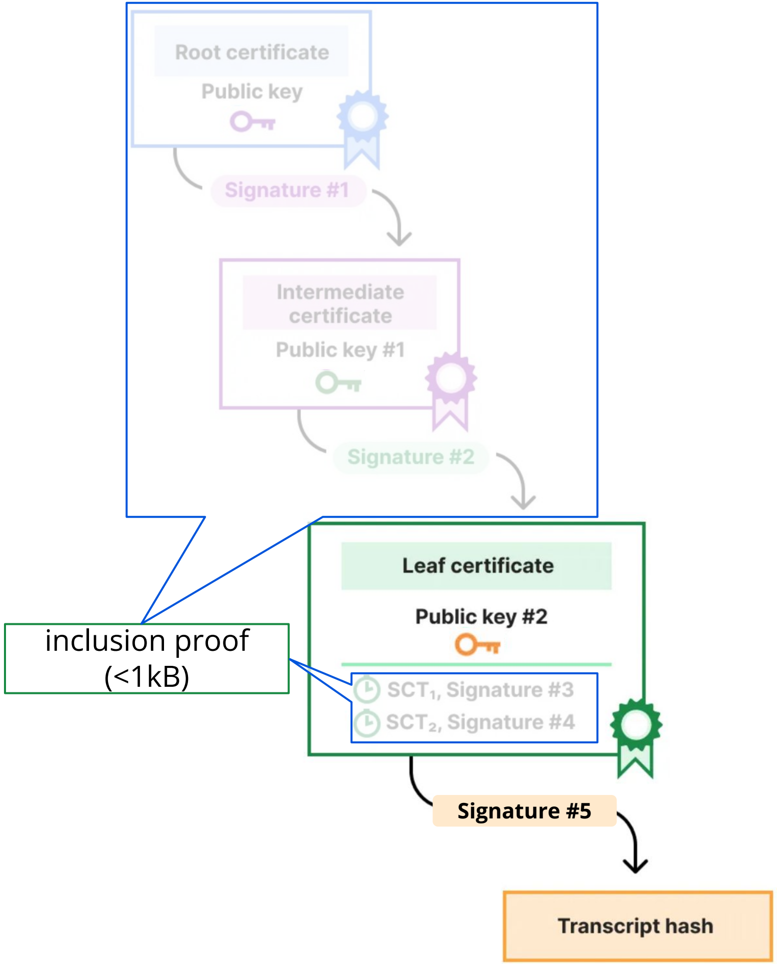
Gist of Merkle Tree Certs

Idea: make the common case fast.

Strategy: Replace entire PKI proof with a single <1kB Merkle Tree inclusion proof.

Scope: Only works for short-lived certificates (7 days, ACME); issued hours in advance; and needs up-to-date relying parties (eg. browsers).

Fall back[1] to X.509+CT for everything else (new domain registration, overlooked cert renewal, unplanned domain move).



[1] Fallbacks needed <0.1% of the time: SCT Auditing Revisited

Signatures 🖋️

Less urgent, but path is unclear. There is a real risk we will start migrating too late. Systemic changes on the table.

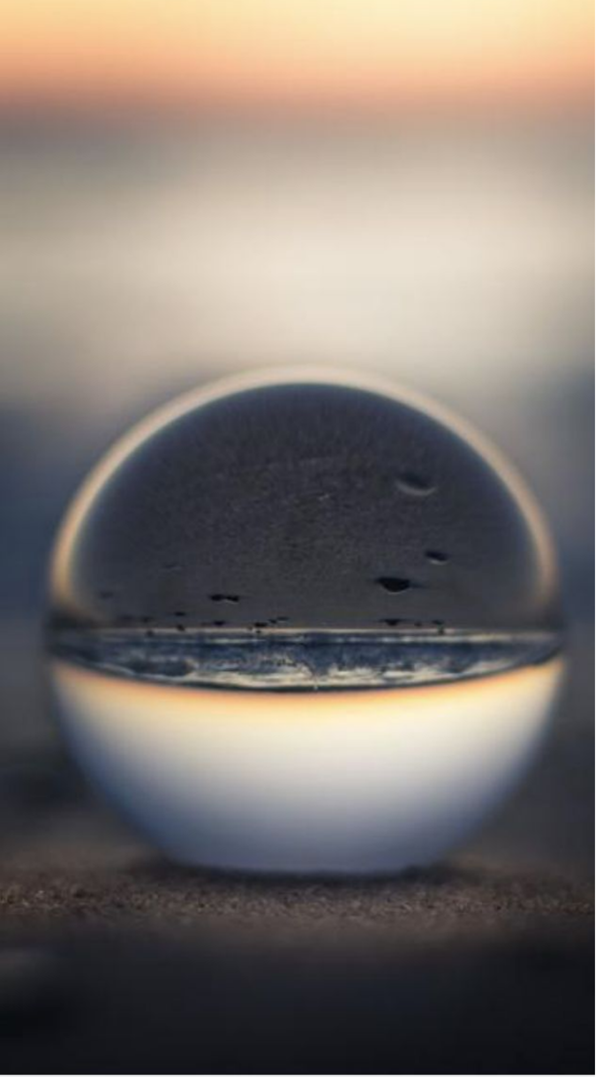
That’s not all: the Internet isn’t just TLS

There is much more cryptography out there with their own unique challenges (eg. DNSSEC).

In particular, for many recent exciting deployment of “fancy” cryptography (anonymous credentials, OPRFs, …), there are no great PQ alternatives.

We’re crowd-sourcing a list here:

github.com/fancy-cryptography/fancy-cryptography



Our plans

We’ll try MTC at scale with Chrome in 2025.

New certificate size experiments to better understand end-user impact.

Preliminary support for PQ certificates to origins in 2025, if standards are ready.

Predictions

PQ key agreement turned on by default in major libraries / platforms by end of 2025.

First drop-in PQ certificate from CAs in 2026, but could slip. Disabled by default.

When do we get a fully PQ Internet?

1. Where are we with the migration?
2. What challenges lay ahead.
3. PQ paradoxes.

We need smaller signatures

We can’t wait for smaller signatures

Every CPU cycle counts

CPU performance is just one consideration

We need options

We can’t deal with options

Silly example: ML-KEM private key format

- FIPS 203 allows seed (d,z) or semi-expanded $(s,t,\rho,H(t,\rho),z)$.
- *Seed* has the edge in size, simplicity (no need to specify consistency checks), and some marginal security properties.
- For compatibility it’s ideal to stick to one format: benefits of seed do not weigh up against supporting two formats…
- After hundreds of emails, IETF ends up with three formats for X.509: seed, expanded, and “both”.
  One cause: preliminary deployment of hardware that doesn’t support seed.
  (Yes, PQC legacy is *already* a thing.)

Mike Ounsworth’s slides

Other examples

- HashML-DSA versus (pure) ML-DSA
- Hashes used within primitives
- Falcon signature format
- Hybrids

Thank you, questions?

ask-research@cloudflare.com