The road ahead to a fully post-quantum Internet

Bas Westerbaan, Cloudflare Research

PQCSA 2026, Brussels, May 19th

About Cloudflare

We run a global network spanning 335 cities in over 125 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 102 million HTTP requests per second.
>42% 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.

“Internet”

Each ecosystem moves at its own pace

The “Internet” website ecosystem has a mixed personality.

- Very diverse in use cases and update cycles. Decentralized. Long tail of legacy clients / servers that will never update but still need to be supported.
- Trailblazer in innovation, also in post-quantum. Seen by many as the gold standard for better or for worse.

What are the steps in the migration?


Algorithm standards
Product support
Deployment

Post-quantum hybrid ECDHE-MLKEM Key Agreement for TLSv1.3

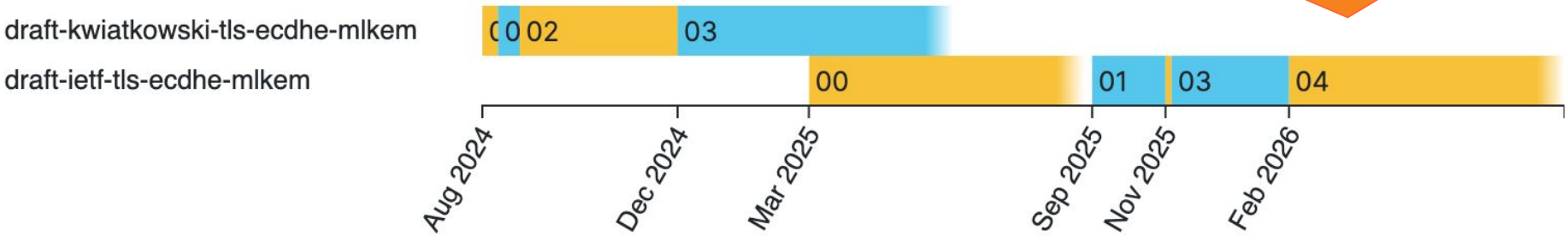
draft-ietf-tls-ecdhe-mlkem-04

Status | IESG evaluation record | IESG writeups | R

Versions:

00 | 01 | 02 | 03 | 04

Expected to be in RFC in a couple of weeks. Good to go, and widely used in production.



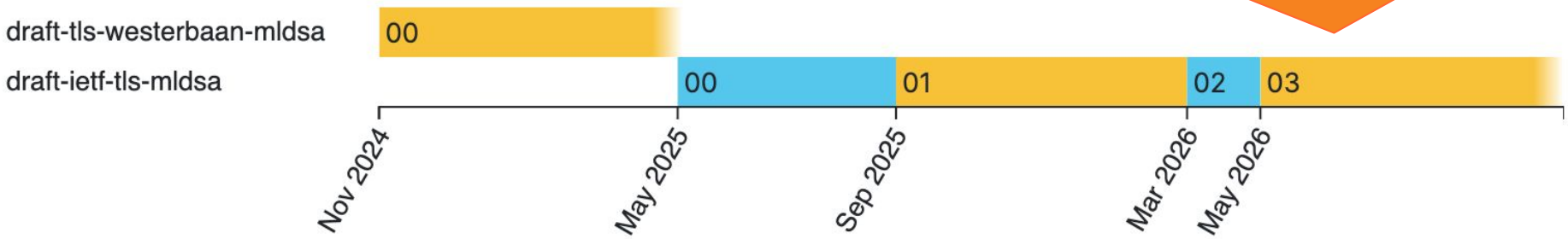
Use of ML-DSA in TLS 1.3

draft-ietf-tls-mldsa-03

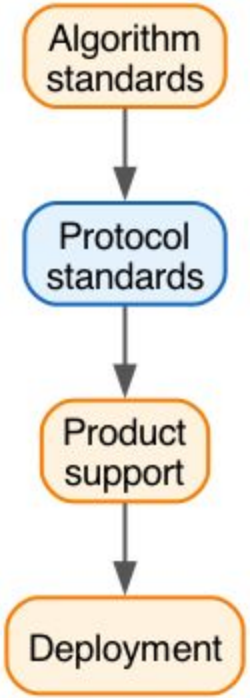
Status | IESG evaluation record | IESG writeups

Versions:

00 | 01 | 02 | 03



Expected to be in RFC in a couple of months. Ready to use in production.


Algorithm standards
Protocol standards
Product support
Deployment

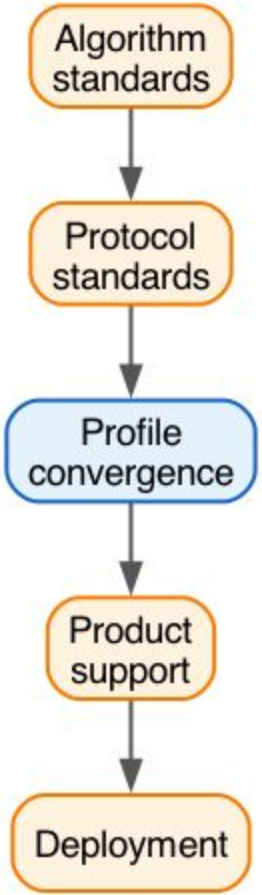
Most systems can’t upgrade all at once

Protocol agility in theory

Alice	Bob	Safe?
Vulnerable	Vulnerable	No
PQ supported Alice: my job is done, just waiting for Bob!	Vulnerable	No
PQ supported	PQ supported Bob: all good!	Yes

First gap: no common algorithm

Alice	Bob	Safe?
Vulnerable	Vulnerable	No
PQ-A supported Alice: my job is done, just waiting for Bob!	Vulnerable	No
PQ-A supported	PQ-B supported Bob: my job is done, just waiting for Alice!	**No!**


Algorithm standards
Protocol standards
Profile convergence
Product support
Deployment

Which algorithm? (TLS certificates)

- MLDSA44-RSA2048-PSS
- MLDSA44-RSA2048-PKCS15
- MLDSA44-Ed25519
- MLDSA44-ECDSA-P256
- MLDSA65-RSA3072-PSS
- MLDSA65-RSA3072-PKCS15
- MLDSA65-RSA4096-PSS
- MLDSA65-RSA4096-PKCS15
- MLDSA65-ECDSA-P256
- MLDSA65-ECDSA-P384
- MLDSA65-ECDSA-brainpoolP256r1
- MLDSA65-Ed25519
- MLDSA87-ECDSA-P384
- MLDSA87-ECDSA-brainpoolP384r1
- MLDSA87-Ed448
- MLDSA87-RSA3072-PSS
- MLDSA87-RSA4096-PSS
- MLDSA87-ECDSA-P521
- MLDSA44
- MLDSA65
- MLDSA87

- slh-dsa-sha2-128s
- slh-dsa-sha2-128f
- slh-dsa-sha2-192s
- slh-dsa-sha2-192f
- slh-dsa-sha2-256s
- slh-dsa-sha2-256f
- slh-dsa-shake-128s
- slh-dsa-shake-128f
- slh-dsa-shake-192s
- slh-dsa-shake-192f
- slh-dsa-shake-256s
- slh-dsa-shake-256f
- hash-slh-dsa-sha2-128s
- hash-slh-dsa-sha2-128f
- hash-slh-dsa-sha2-192s
- hash-slh-dsa-sha2-192f
- hash-slh-dsa-sha2-256s
- hash-slh-dsa-sha2-256f
- hash-slh-dsa-shake-128s
- hash-slh-dsa-shake-128f
- hash-slh-dsa-shake-192s
- hash-slh-dsa-shake-192f
- hash-slh-dsa-shake-256s
- hash-slh-dsa-shake-256f

Many servers struggle installing even *two* certificates.

My thought process (1)

- MLDSA44-RSA2048-PSS
- MLDSA44-RSA2048-PKCS15
- MLDSA44-Ed25519
- MLDSA44-ECDSA-P256
- MLDSA65-RSA3072-PSS
- MLDSA65-RSA3072-PKCS15
- MLDSA65-RSA4096-PSS
- MLDSA65-RSA4096-PKCS15
- MLDSA65-ECDSA-P256
- MLDSA65-ECDSA-P384
- MLDSA65-ECDSA-brainpoolP256r1
- MLDSA65-Ed25519
- MLDSA87-ECDSA-P384
- MLDSA87-ECDSA-brainpoolP384r1
- MLDSA87-Ed448
- MLDSA87-RSA3072-PSS
- MLDSA87-RSA4096-PSS
- MLDSA87-ECDSA-P521
- MLDSA44
- MLDSA65
- MLDSA87

- ~~slh-dsa-sha2-128s~~
- ~~slh-dsa-sha2-128f~~
- ~~slh-dsa-sha2-192s~~
- ~~slh-dsa-sha2-192f~~
- ~~slh-dsa-sha2-256s~~
- ~~slh-dsa-sha2-256f~~
- ~~slh-dsa-shake-128s~~
- ~~slh-dsa-shake-128f~~
- ~~slh-dsa-shake-192s~~
- ~~slh-dsa-shake-192f~~
- ~~slh-dsa-shake-256s~~
- ~~slh-dsa-shake-256f~~
- ~~hash-slh-dsa-sha2-128s~~
- ~~hash-slh-dsa-sha2-128f~~
- ~~hash-slh-dsa-sha2-192s~~
- ~~hash-slh-dsa-sha2-192f~~
- ~~hash-slh-dsa-sha2-256s~~
- ~~hash-slh-dsa-sha2-256f~~
- ~~hash-slh-dsa-shake-128s~~
- ~~hash-slh-dsa-shake-128f~~
- ~~hash-slh-dsa-shake-192s~~
- ~~hash-slh-dsa-shake-192f~~
- ~~hash-slh-dsa-shake-256s~~
- ~~hash-slh-dsa-shake-256f~~

We need ML-KEM to be secure today, so relying on ML-DSA doesn’t add much of a security assumption.

My thought process (2)

- MLDSA44-RSA2048-PSS
- MLDSA44-RSA2048-PKCS15
- MLDSA44-Ed25519
- MLDSA44-ECDSA-P256
- ~~**MLDSA65-RSA3072-PSS**~~
- ~~**MLDSA65-RSA3072-PKCS15**~~
- ~~**MLDSA65-RSA4096-PSS**~~
- ~~**MLDSA65-RSA4096-PKCS15**~~
- ~~**MLDSA65-ECDSA-P256**~~
- ~~**MLDSA65-ECDSA-P384**~~
- ~~**MLDSA65-ECDSA-brainpoolP256r1**~~
- ~~**MLDSA65-Ed25519**~~
- ~~**MLDSA87-ECDSA-P384**~~
- ~~**MLDSA87-ECDSA-brainpoolP384r1**~~
- ~~**MLDSA87-Ed448**~~
- ~~**MLDSA87-RSA3072-PSS**~~
- ~~**MLDSA87-RSA4096-PSS**~~
- ~~**MLDSA87-ECDSA-P521**~~
- MLDSA44
- ~~**MLDSA65**~~
- ~~**MLDSA87**~~

- ~~slh-dsa-sha2-128s~~
- ~~slh-dsa-sha2-128f~~
- ~~slh-dsa-sha2-192s~~
- ~~slh-dsa-sha2-192f~~
- ~~slh-dsa-sha2-256s~~
- ~~slh-dsa-sha2-256f~~
- ~~slh-dsa-shake-128s~~
- ~~slh-dsa-shake-128f~~
- ~~slh-dsa-shake-192s~~
- ~~slh-dsa-shake-192f~~
- ~~slh-dsa-shake-256s~~
- ~~slh-dsa-shake-256f~~
- ~~hash-slh-dsa-sha2-128s~~
- ~~hash-slh-dsa-sha2-128f~~
- ~~hash-slh-dsa-sha2-192s~~
- ~~hash-slh-dsa-sha2-192f~~
- ~~hash-slh-dsa-sha2-256s~~
- ~~hash-slh-dsa-sha2-256f~~
- ~~hash-slh-dsa-shake-128s~~
- ~~hash-slh-dsa-shake-128f~~
- ~~hash-slh-dsa-shake-192s~~
- ~~hash-slh-dsa-shake-192f~~
- ~~hash-slh-dsa-shake-256s~~
- ~~hash-slh-dsa-shake-256f~~

Ciphertext needs to be secure forever. Certificates can be rolled when they’re weakened. ML-DSA-44 has a very comfortable margin.

My thought process (3)

- ~~**MLDSA44-RSA2048-PSS**~~
- ~~**MLDSA44-RSA2048-PKCS15**~~
- MLDSA44-Ed25519
- MLDSA44-ECDSA-P256
- ~~MLDSA65-RSA3072-PSS~~
- ~~MLDSA65-RSA3072-PKCS15~~
- ~~MLDSA65-RSA4096-PSS~~
- ~~MLDSA65-RSA4096-PKCS15~~
- ~~MLDSA65-ECDSA-P256~~
- ~~MLDSA65-ECDSA-P384~~
- ~~MLDSA65-ECDSA-brainpoolP256r1~~
- ~~MLDSA65-Ed25519~~
- ~~MLDSA87-ECDSA-P384~~
- ~~MLDSA87-ECDSA-brainpoolP384r1~~
- ~~MLDSA87-Ed448~~
- ~~MLDSA87-RSA3072-PSS~~
- ~~MLDSA87-RSA4096-PSS~~
- ~~MLDSA87-ECDSA-P521~~
- MLDSA44
- ~~MLDSA65~~
- ~~MLDSA87~~

- ~~slh-dsa-sha2-128s~~
- ~~slh-dsa-sha2-128f~~
- ~~slh-dsa-sha2-192s~~
- ~~slh-dsa-sha2-192f~~
- ~~slh-dsa-sha2-256s~~
- ~~slh-dsa-sha2-256f~~
- ~~slh-dsa-shake-128s~~
- ~~slh-dsa-shake-128f~~
- ~~slh-dsa-shake-192s~~
- ~~slh-dsa-shake-192f~~
- ~~slh-dsa-shake-256s~~
- ~~slh-dsa-shake-256f~~
- ~~hash-slh-dsa-sha2-128s~~
- ~~hash-slh-dsa-sha2-128f~~
- ~~hash-slh-dsa-sha2-192s~~
- ~~hash-slh-dsa-sha2-192f~~
- ~~hash-slh-dsa-sha2-256s~~
- ~~hash-slh-dsa-sha2-256f~~
- ~~hash-slh-dsa-shake-128s~~
- ~~hash-slh-dsa-shake-128f~~
- ~~hash-slh-dsa-shake-192s~~
- ~~hash-slh-dsa-shake-192f~~
- ~~hash-slh-dsa-shake-256s~~
- ~~hash-slh-dsa-shake-256f~~

RSA's only remaining benefit is fast verification time, which is negated by combining with ML-DSA-44's, which has verification times on the order of EC.

My thought process (4)

- ~~MLDSA44-RSA2048-PSS~~
- ~~MLDSA44-RSA2048-PKCS15~~
- ~~**MLDSA44-Ed25519**~~
- MLDSA44-ECDSA-P256
- ~~MLDSA65-RSA3072-PSS~~
- ~~MLDSA65-RSA3072-PKCS15~~
- ~~MLDSA65-RSA4096-PSS~~
- ~~MLDSA65-RSA4096-PKCS15~~
- ~~MLDSA65-ECDSA-P256~~
- ~~MLDSA65-ECDSA-P384~~
- ~~MLDSA65-ECDSA-brainpoolP256r1~~
- ~~MLDSA65-Ed25519~~
- ~~MLDSA87-ECDSA-P384~~
- ~~MLDSA87-ECDSA-brainpoolP384r1~~
- ~~MLDSA87-Ed448~~
- ~~MLDSA87-RSA3072-PSS~~
- ~~MLDSA87-RSA4096-PSS~~
- ~~MLDSA87-ECDSA-P521~~
- MLDSA44
- ~~MLDSA65~~
- ~~MLDSA87~~

- ~~slh-dsa-sha2-128s~~
- ~~slh-dsa-sha2-128f~~
- ~~slh-dsa-sha2-192s~~
- ~~slh-dsa-sha2-192f~~
- ~~slh-dsa-sha2-256s~~
- ~~slh-dsa-sha2-256f~~
- ~~slh-dsa-shake-128s~~
- ~~slh-dsa-shake-128f~~
- ~~slh-dsa-shake-192s~~
- ~~slh-dsa-shake-192f~~
- ~~slh-dsa-shake-256s~~
- ~~slh-dsa-shake-256f~~
- ~~hash-slh-dsa-sha2-128s~~
- ~~hash-slh-dsa-sha2-128f~~
- ~~hash-slh-dsa-sha2-192s~~
- ~~hash-slh-dsa-sha2-192f~~
- ~~hash-slh-dsa-sha2-256s~~
- ~~hash-slh-dsa-sha2-256f~~
- ~~hash-slh-dsa-shake-128s~~
- ~~hash-slh-dsa-shake-128f~~
- ~~hash-slh-dsa-shake-192s~~
- ~~hash-slh-dsa-shake-192f~~
- ~~hash-slh-dsa-shake-256s~~
- ~~hash-slh-dsa-shake-256f~~

Even though Ed25519 has advantages, it's regrettably not used in the WebPKI today. Let's not impose unnecessary extra work.

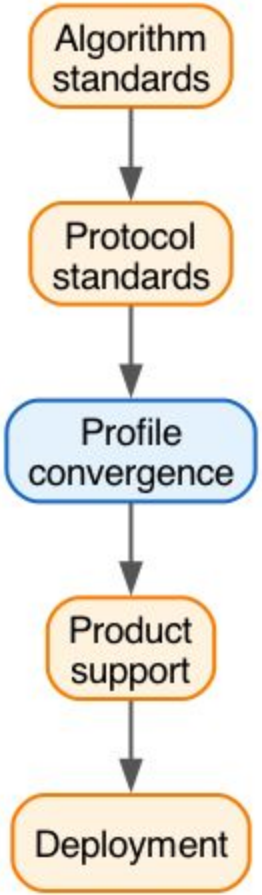
My thought process (5)

That leaves MLDSA44-ECDSA-P256 and ML-DSA-44.

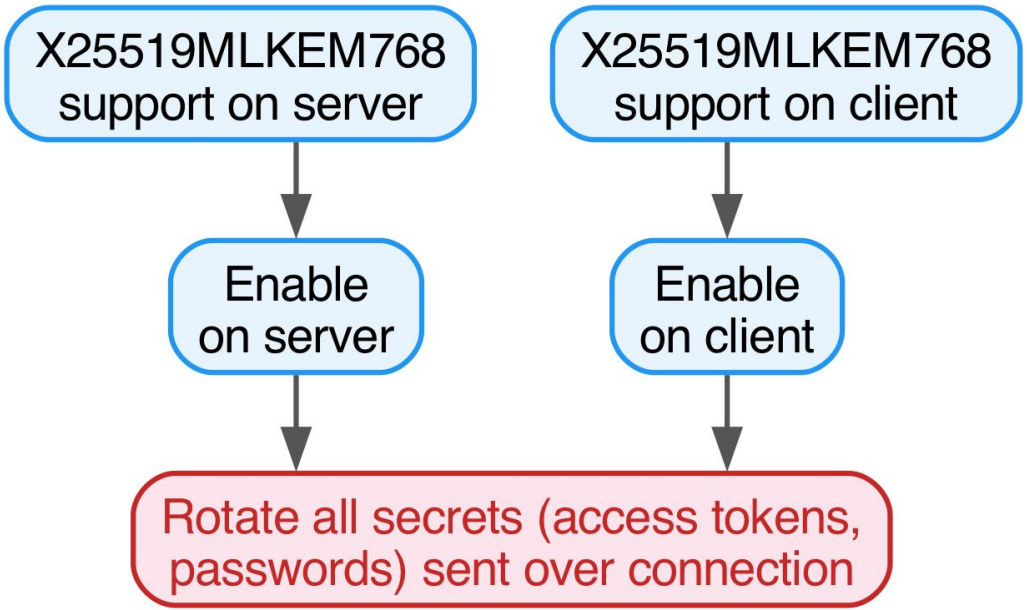
I prefer the hybrid, as it can be deployed without *having to convince anyone*.

Unfortunately it is clear there is no consensus that this particular hybrid should be the go-to like X25519MLKEM768 is for key agreement in the WebPKI.

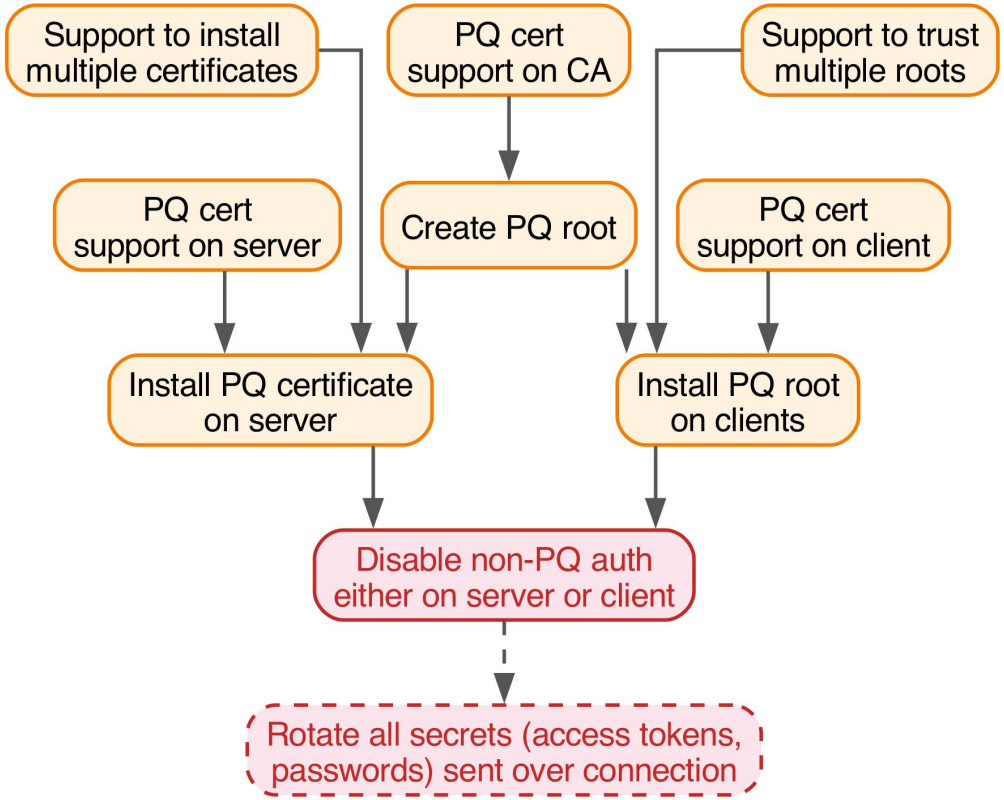
For maximum compatibility, as it stands, ML-DSA-44 is the safest bet.

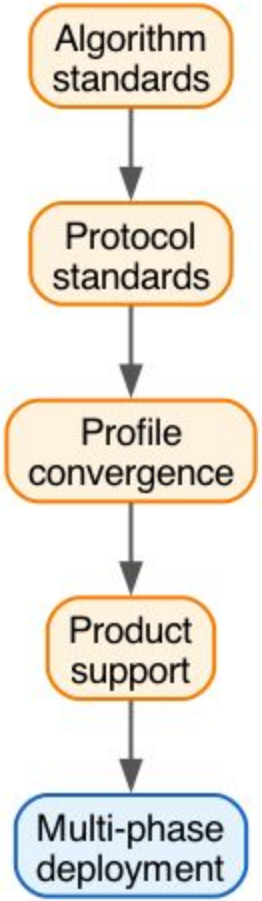

Algorithm standards
Protocol standards
Profile convergence
Product support
Deployment

Mitigating HN/DL is “easy”



Post-quantum auth is a tad more involved



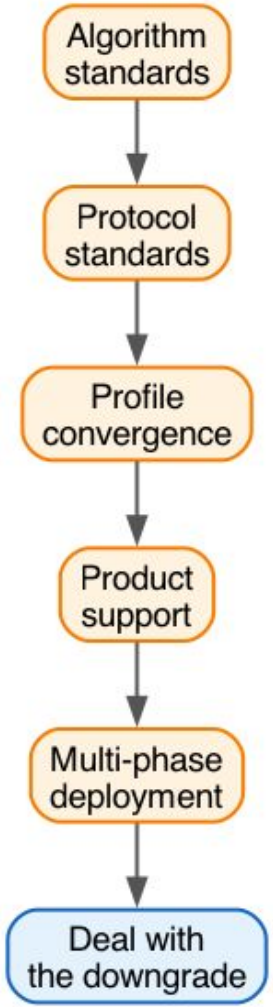

Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment

Protocol agility in theory

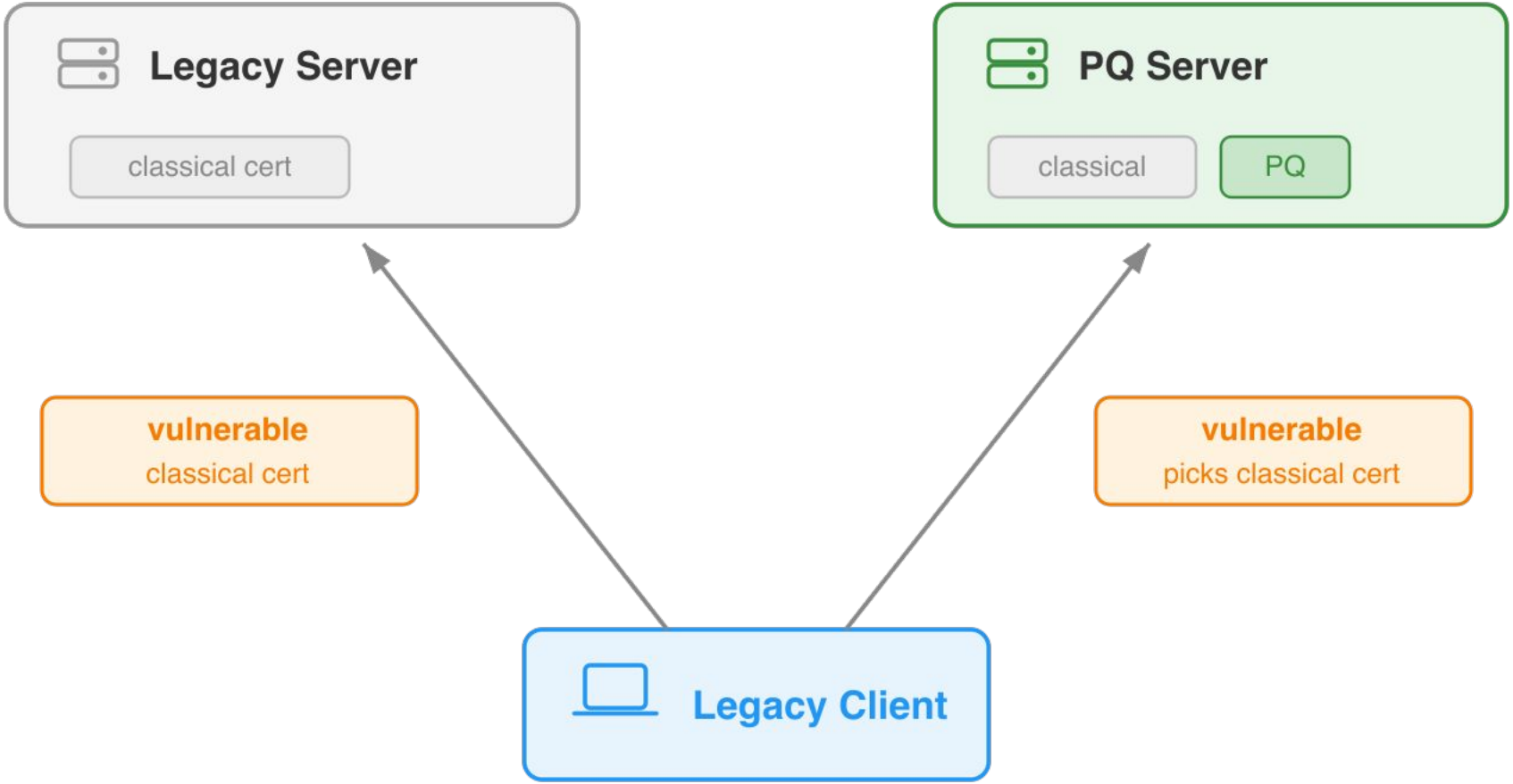
Alice	Bob	Safe?
Vulnerable	Vulnerable	No
PQ supported Alice: my job is done, just waiting for Bob!	Vulnerable	No
PQ supported	PQ supported Bob: all good!	Yes

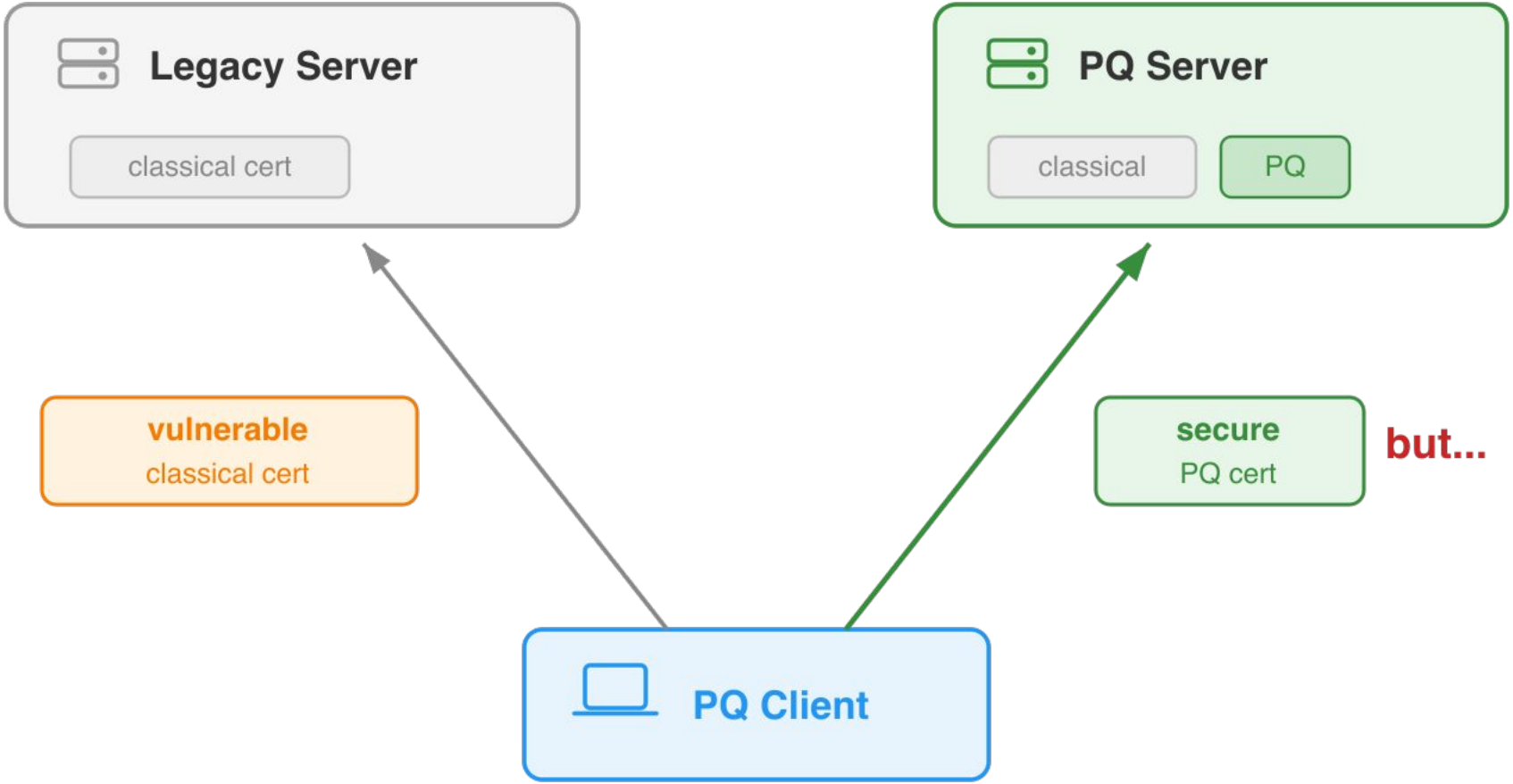
Second gap: downgrades

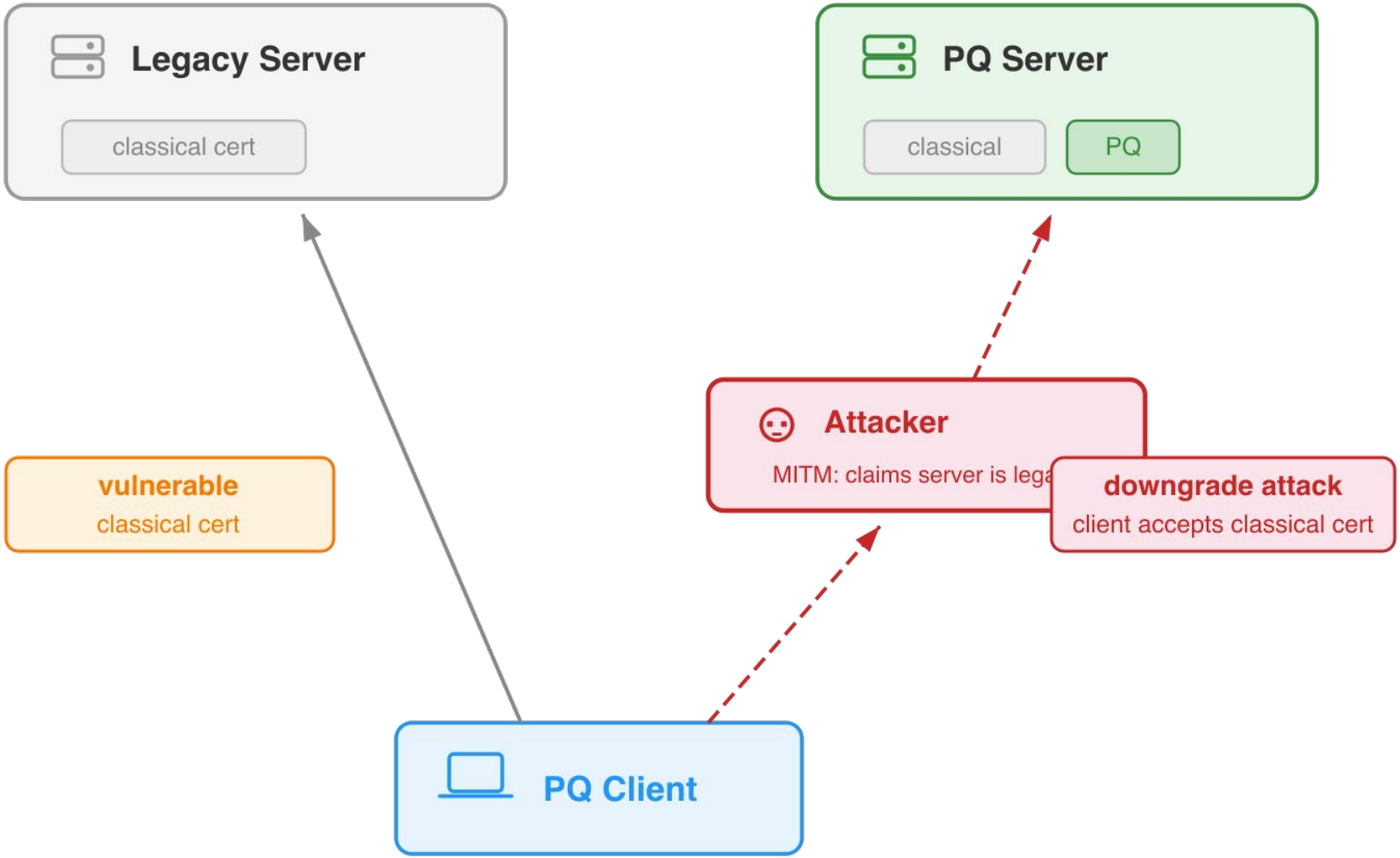
Alice	Bob	Safe?
Vulnerable	Vulnerable	No
PQ & vulnerable	Vulnerable	No
PQ & vulnerable	PQ & vulnerable	**No!**


Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade

Turning off support for vulnerable cryptography on client and server prevents downgrades, but in most large systems this is simply impossible.


Legacy Server
classical cert
PQ Server
classical
PQ
vulnerable
classical cert
vulnerable
picks classical cert
Legacy Client


Legacy Server
classical cert
PQ Server
classical
PQ
vulnerable
classical cert
secure
PQ cert
but...
PQ Client


Legacy Server
classical cert
PQ Server
classical
PQ
Attacker
MITM: claims server is lega
vulnerable
classical cert
downgrade attack
client accepts classical cert
PQ Client

Preventing downgrades with PQ-HSTS

Server can signal to a client that it will have a post-quantum certificate going forward. This is similar to how today HSTS signals that a server will support TLS.

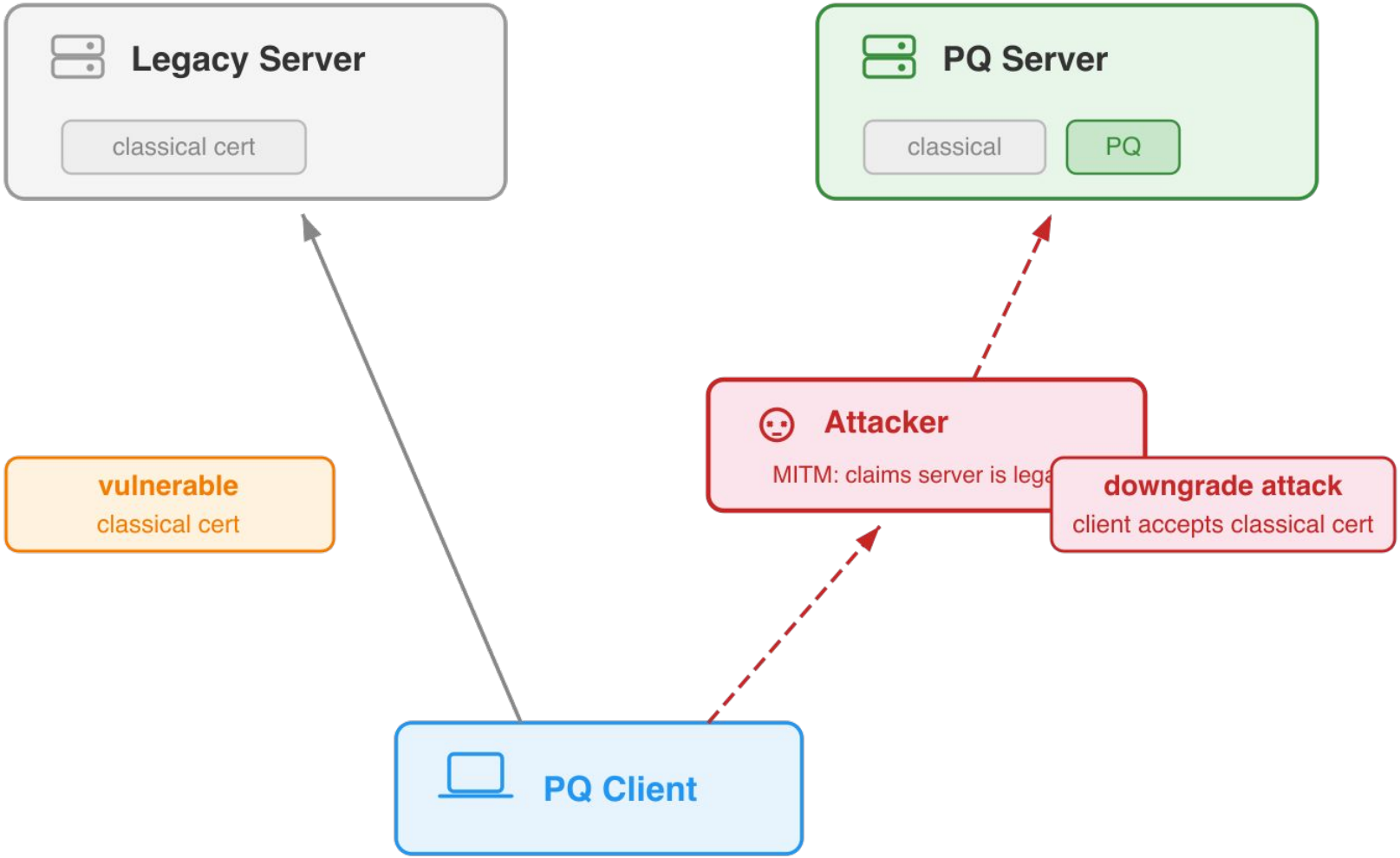
Downside is that this doesn’t work for first connections and can cause surprise breakages when switching to a provider that doesn’t support PQ certs.

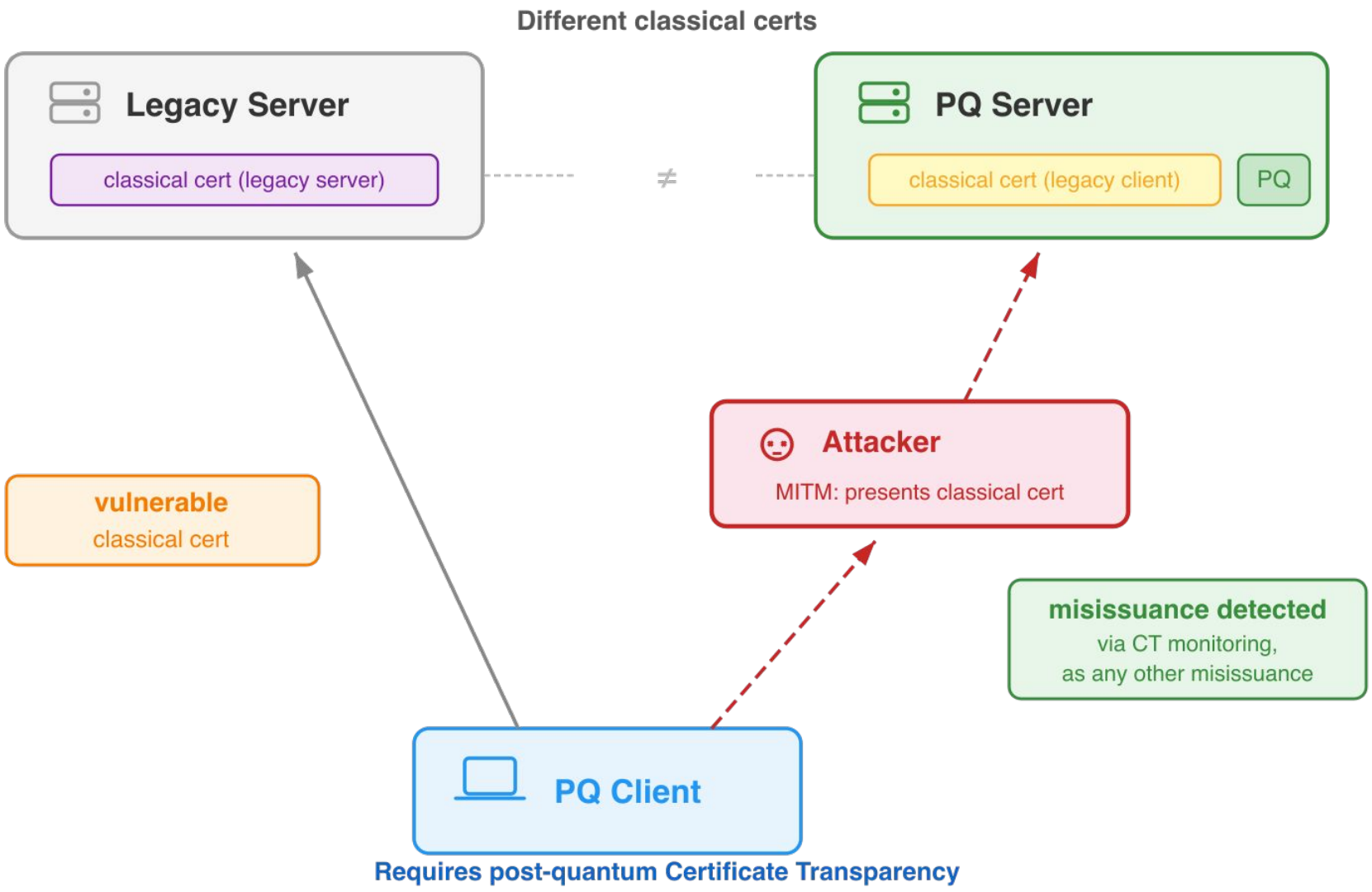
Detecting downgrades using transparency

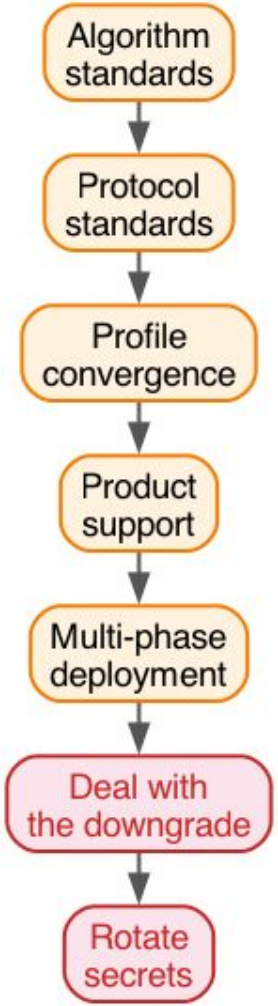
Key idea: distinguish between the certificate chain for the legacy server and legacy client.

Then we can detect issuance of a *downgrade certificate* for legacy servers, using Certificate Transparency.

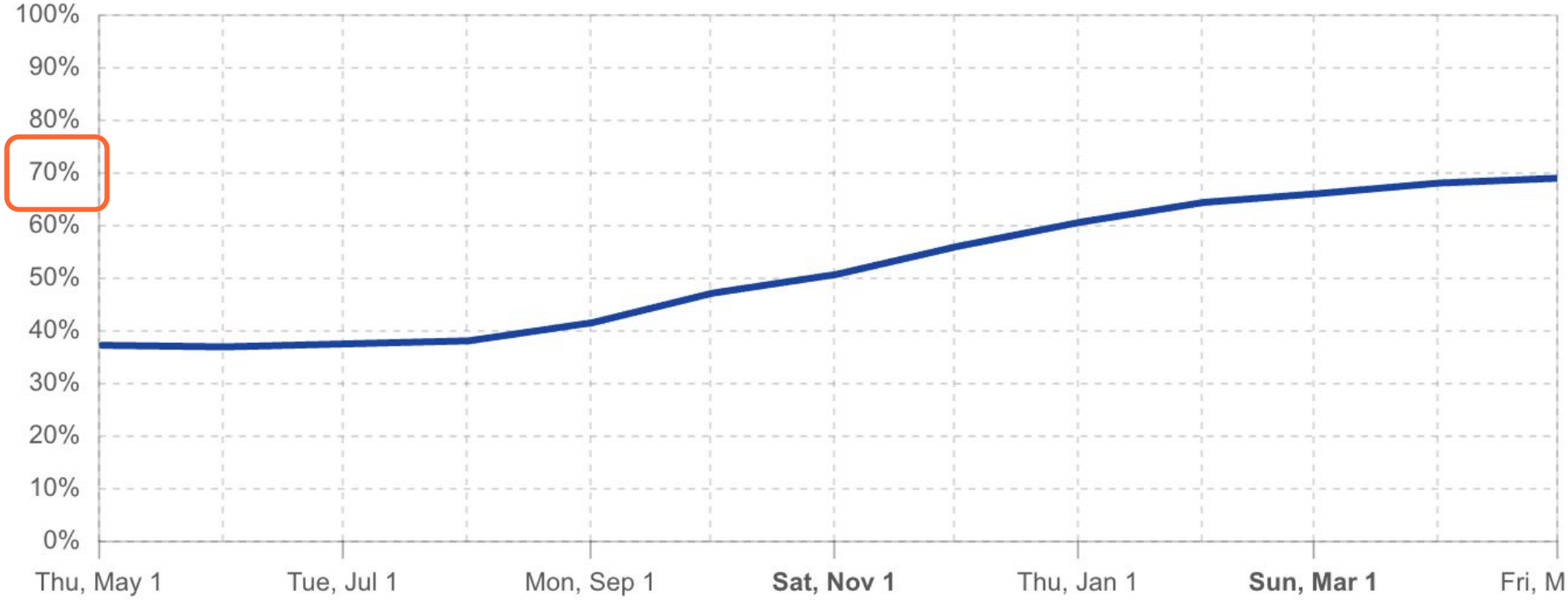
This of course requires the transparency to be post-quantum, like with Merkle Tree Certificates. See also Chrome's authentication roadmap.


Legacy Server
classical cert
PQ Server
classical
PQ
Attacker
MITM: claims server is lega
vulnerable
classical cert
downgrade attack
client accepts classical cert
PQ Client


Different classical certs
Legacy Server
classical cert (legacy server)
≠
PQ Server
classical cert (legacy client)
PQ
Attacker
MITM: presents classical cert
vulnerable
classical cert
misissuance detected
via CT monitoring,
as any other misissuance
PQ Client
Requires post-quantum Certificate Transparency

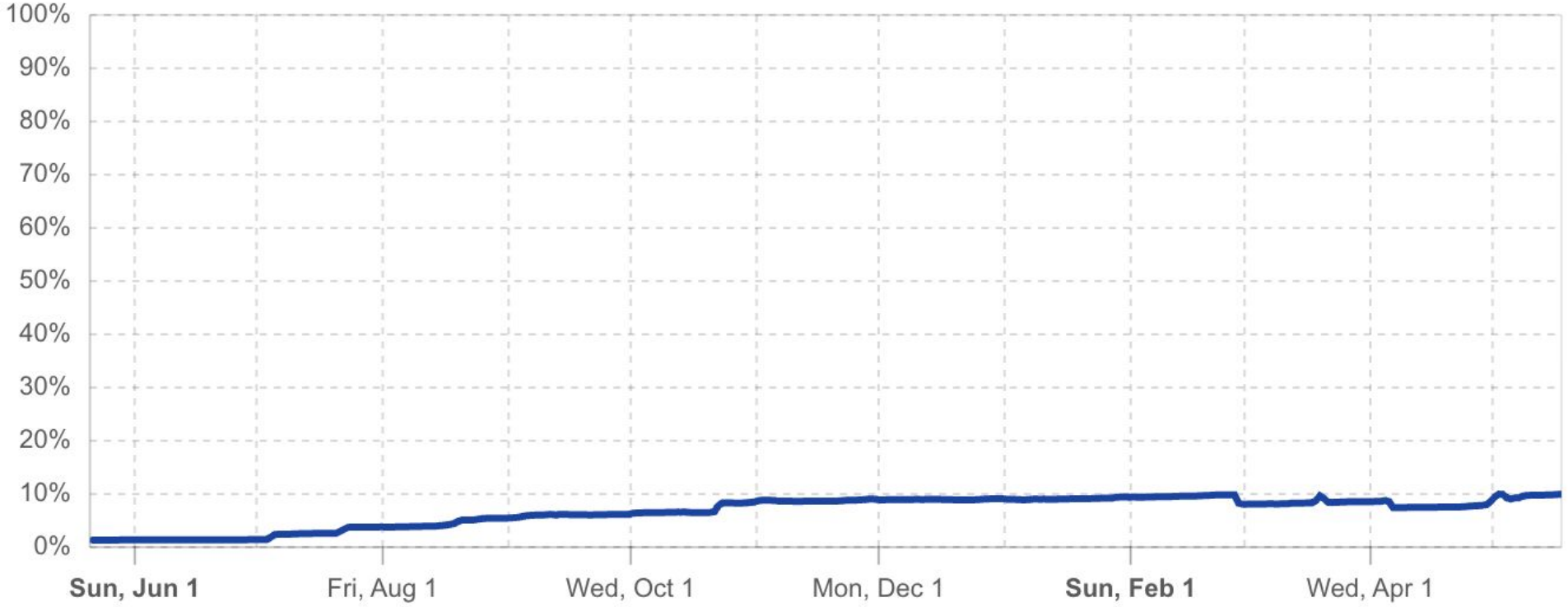

Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade
Rotate secrets

Cloudflare visitor support for X25519MLKEM768



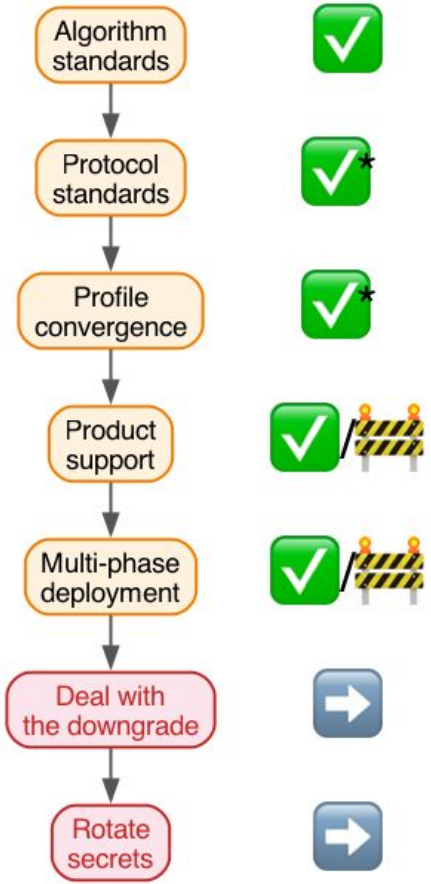
May 2025–2026 from radar.cloudflare.com/post-quantum

Support at Cloudflare customer’s origins



May 2025–2026 from radar.cloudflare.com/post-quantum

HNDL


Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade
Rotate secrets

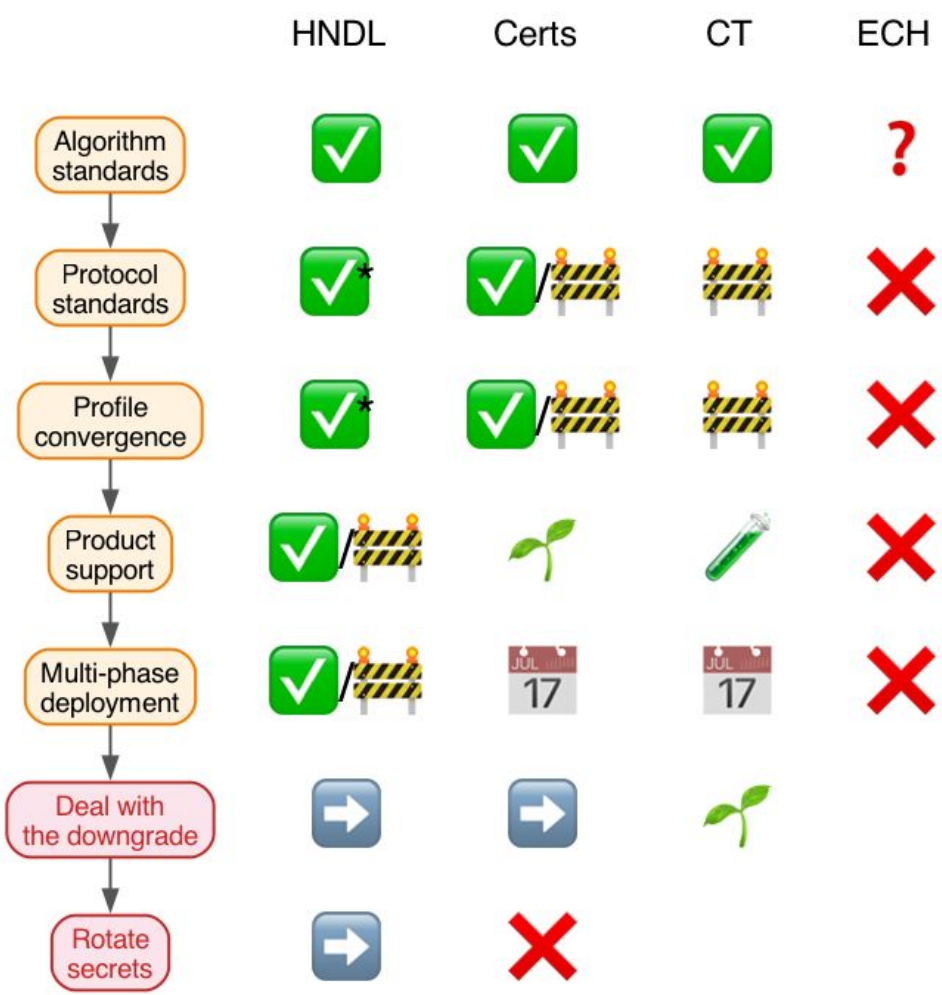
Post-quantum certificate deployment in the WebPKI?

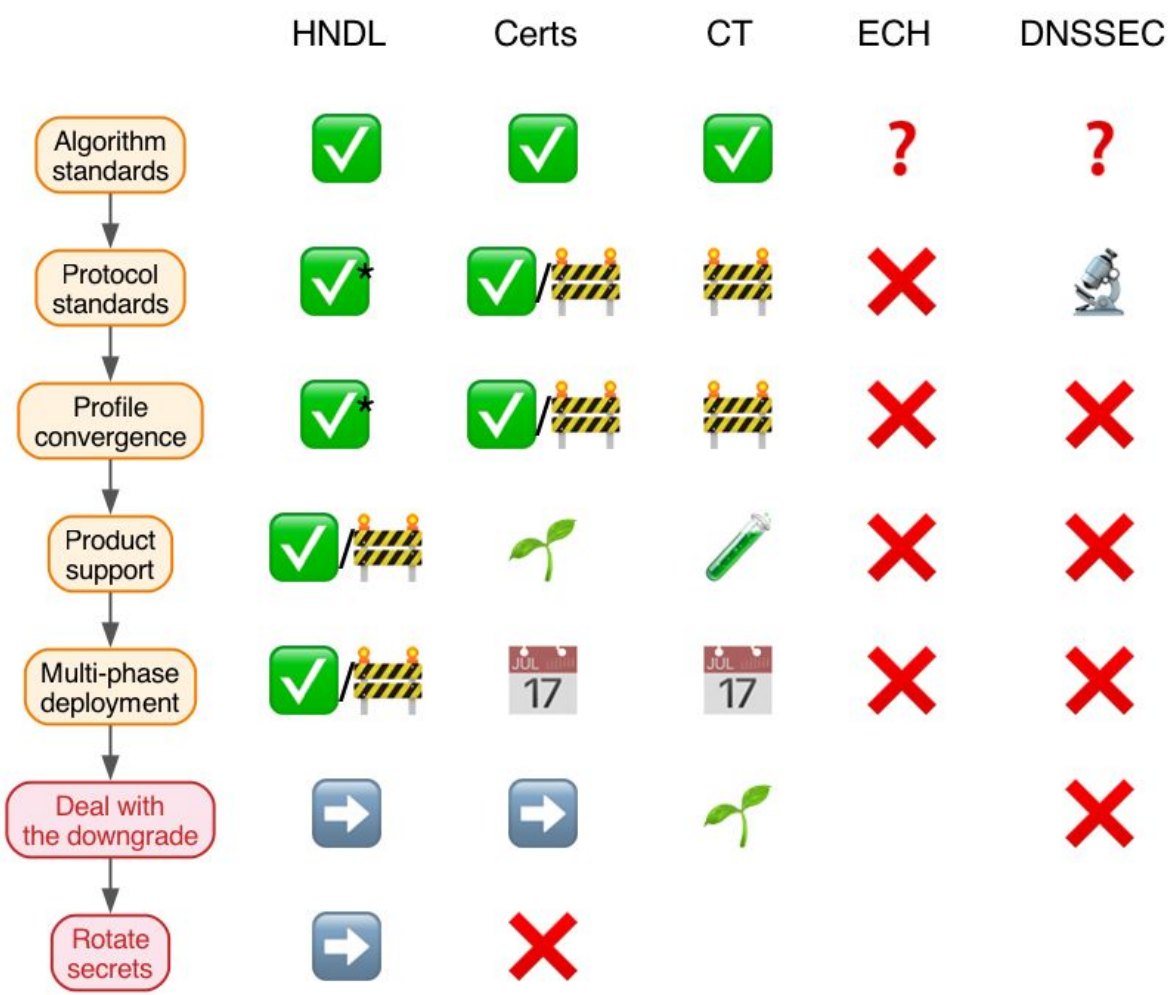
(nothing)

… but widespread deployment coming early 2027.

We’re in the experimental phase now.


HNDL
Certs
CT
Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade
Rotate secrets
JUL
17
JUL
17


HNDL
Certs
CT
ECH
Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade
Rotate secrets
JUL
17
JUL
17


HNDL
Certs
CT
ECH
DNSSEC
Algorithm standards
Protocol standards
Profile convergence
Product support
Multi-phase deployment
Deal with the downgrade
Rotate secrets
JUL
17
JUL
17

What should organizations do now?

Deploy where available—or even just a pilot.

Best practices help: keep software up-to-date, use modern protocols, automated rotation of secrets and issuance.

But first and foremost:

- Determine the business continuity impact of a quantum emergency. This is not a CBOM.
- Think of (non-cryptographic) mitigations.
- Prepare a quantum emergency recovery playbook.

In summary

- We have the NIST standards. They'll have to do.
- Agility: you surface technical gaps by trying.
- The more federated an ecosystem is, the more important it is to pick a small set of common algorithms.
- You're not done until you've dealt with downgrades!
- We need recovery playbooks, not inventories.

Thank you, questions?

blog.cloudflare.com/post-quantum-roadmap