

Get in Loser, We're Upgrading the Internet!

Deploying Post-Quantum Cryptography across Akamai's global CDN

Jan Schaumann <jschauma@akamai.com>
Chief Architect, Information Security

2026-06-24

TROOPERS26

Agenda

Wear pink.
(It's Wednesday.)

Quantum Computing vs
Quantum Cryptography vs
Post-Quantum Cryptography

PQC Threat Model
and shifting priorities

Akamai
PQC Program
and Lessons



NO “Step 1: build your cryptographic inventory”

NO doomsday scenarios around CRQC imminence

NO conspiracy theories around hybrid vs pure

**Guaranteed,
abso-effin-lutely
NO AI**

Akamai - we make fetch happen



- All top 10 video streaming services
- All top 10 video game companies
- All top 10 brokerages
- All top 10 banking companies
- 9 of the top 10 telecommunications carriers
- 8 of the top 10 software companies
- 8 of the top 10 healthcare providers
- 8 of the top 10 automotive companies
- 8 of the top 10 fintech companies
- 8 of the top 10 retail companies
- 7 of the top 10 healthcare payers
- 7 of the top 10 pharma companies
- All 6 U.S. military branches
- 14 of 15 U.S. federal civilian cabinet agencies



Core Compute Regions



Distributed PoPs

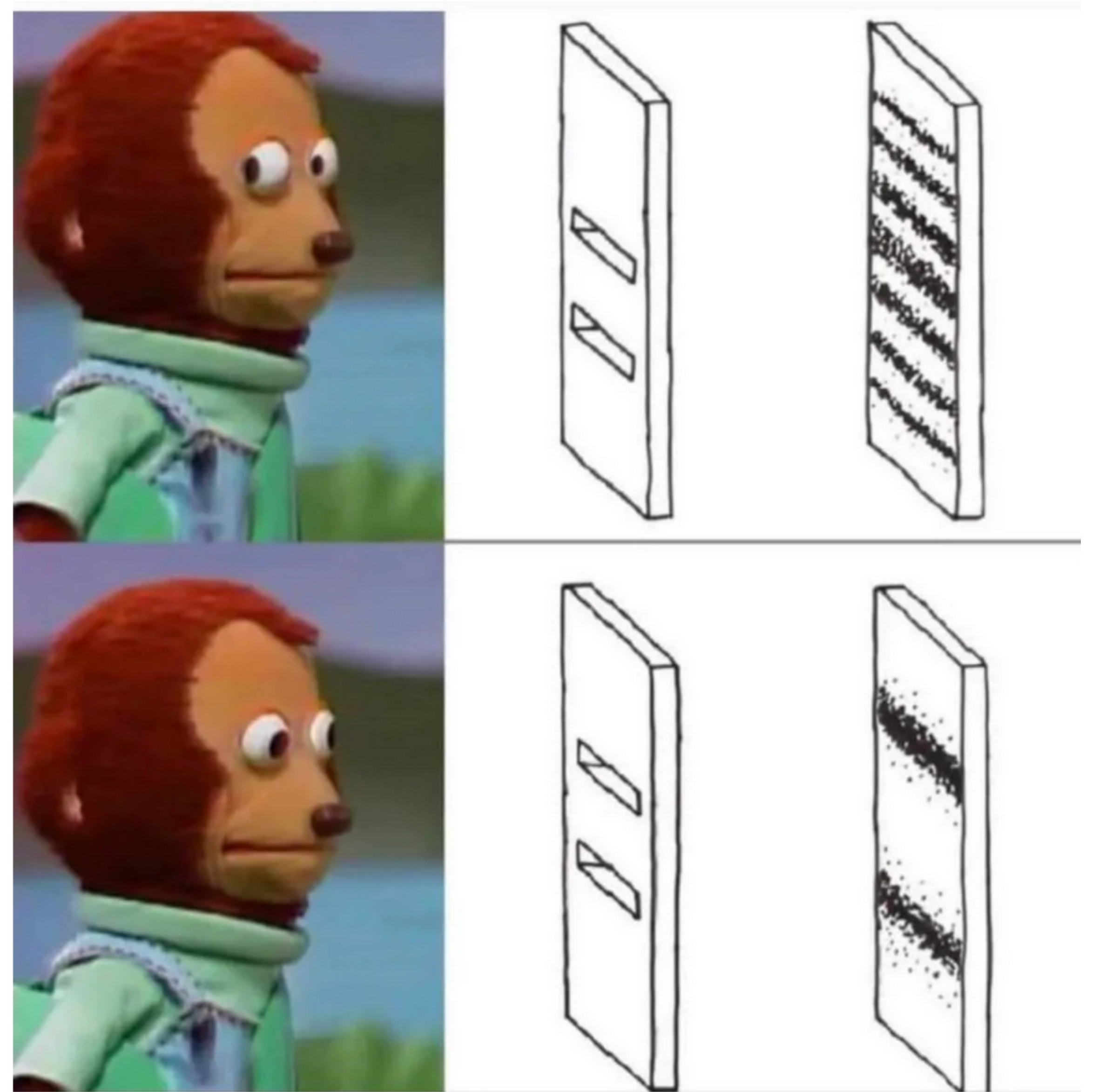


Edge PoPs

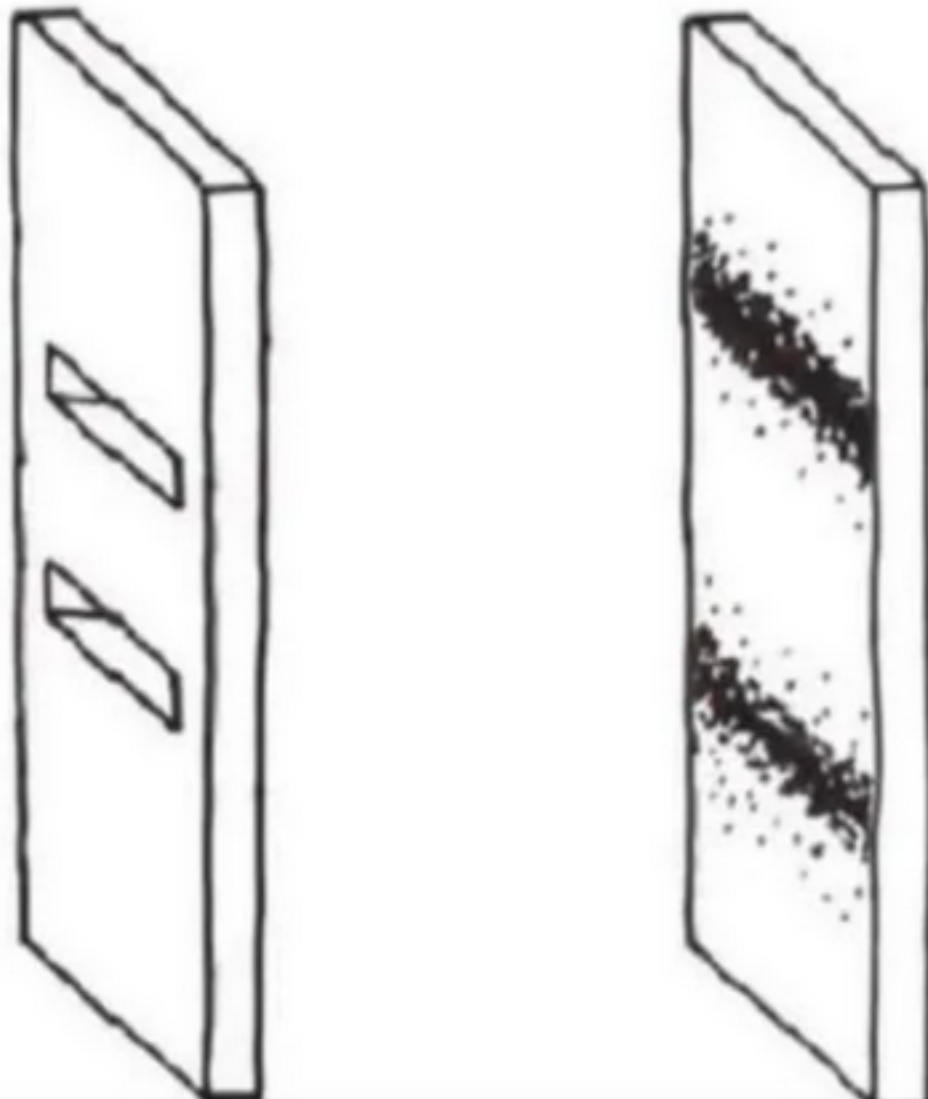
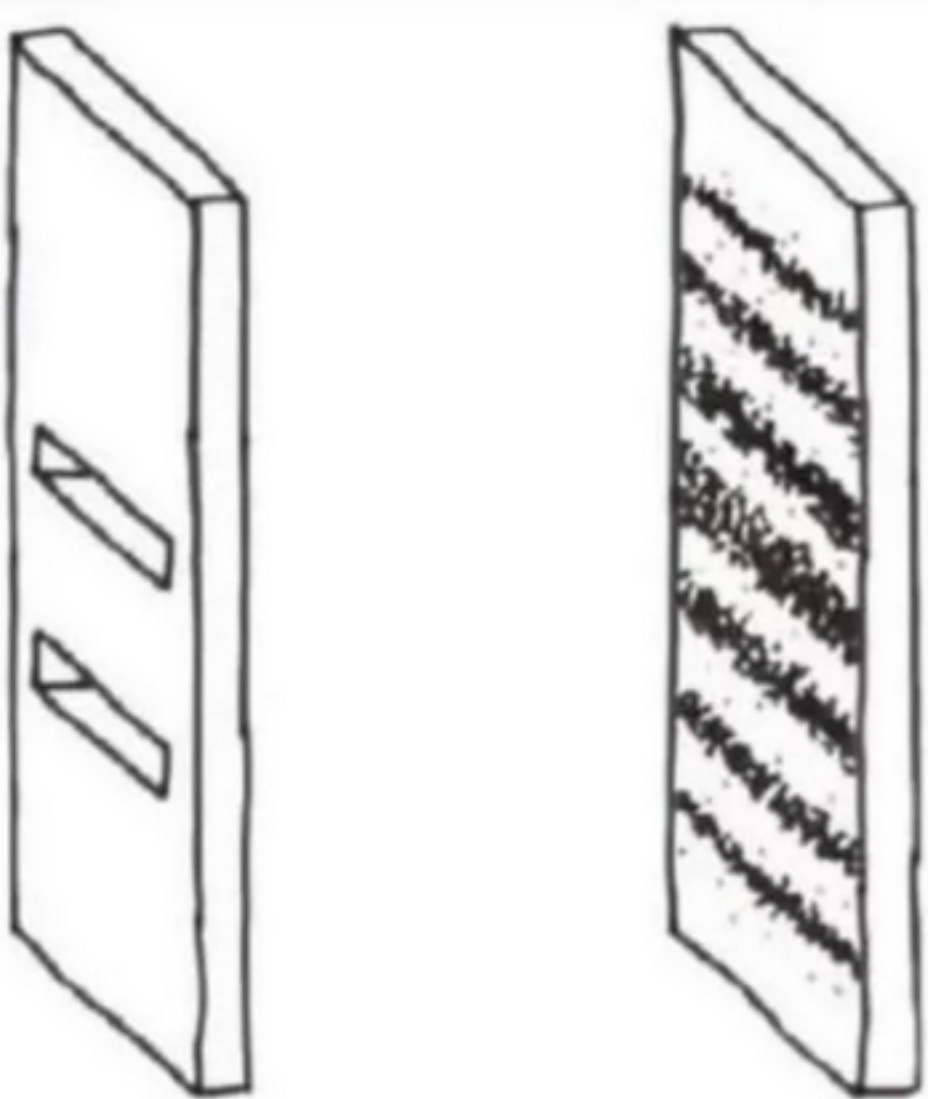
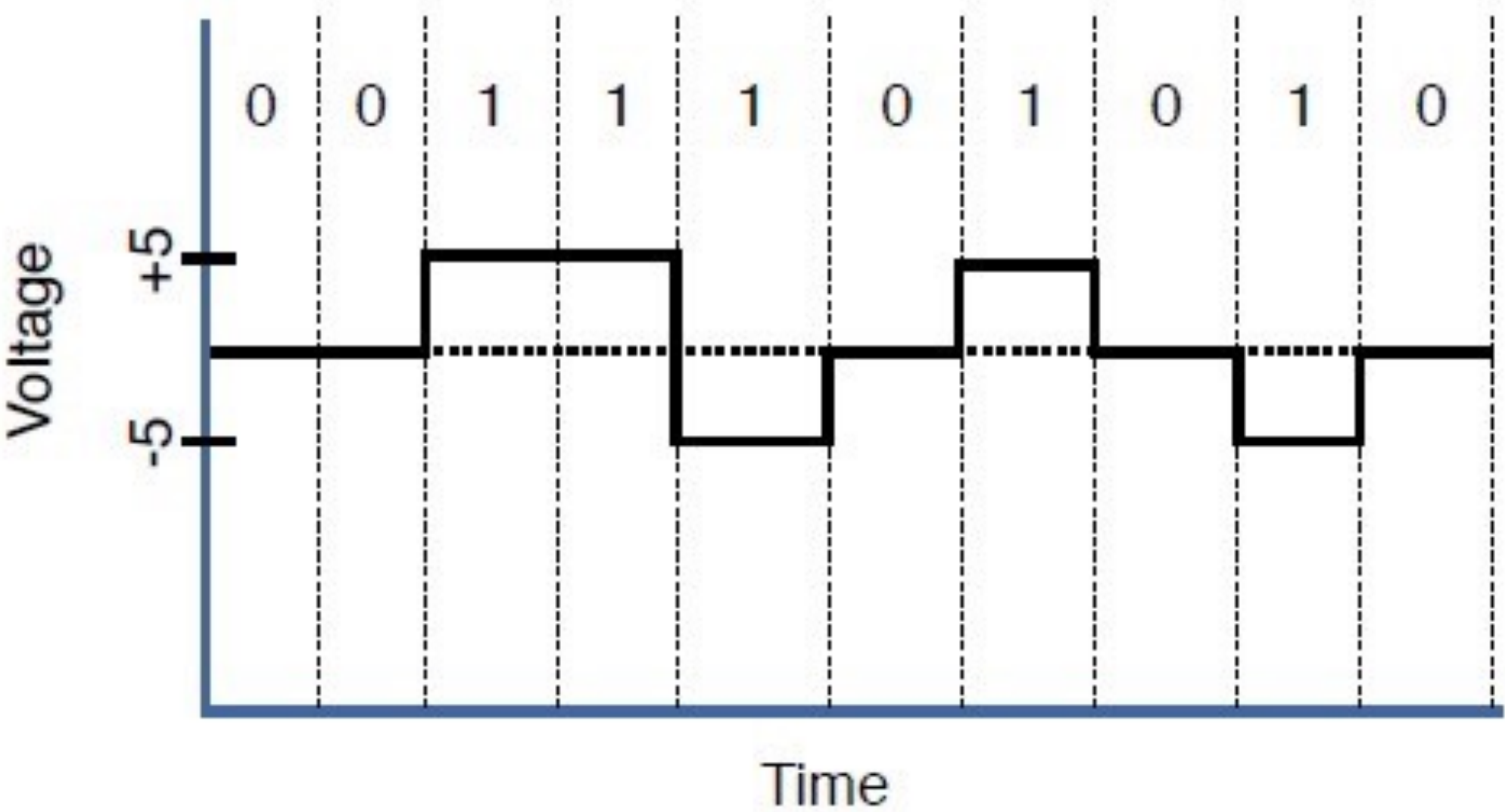
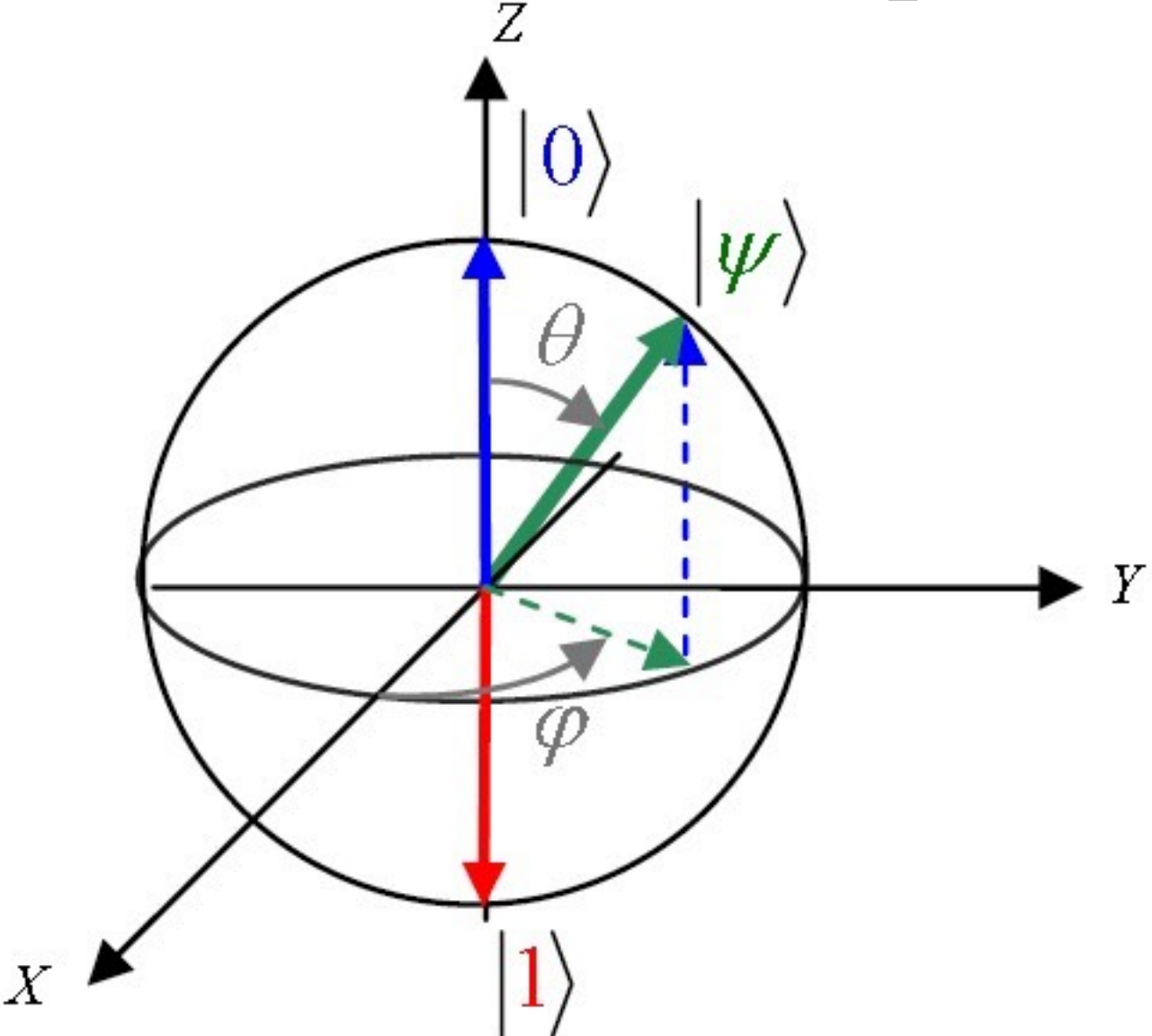


Quantum Computing

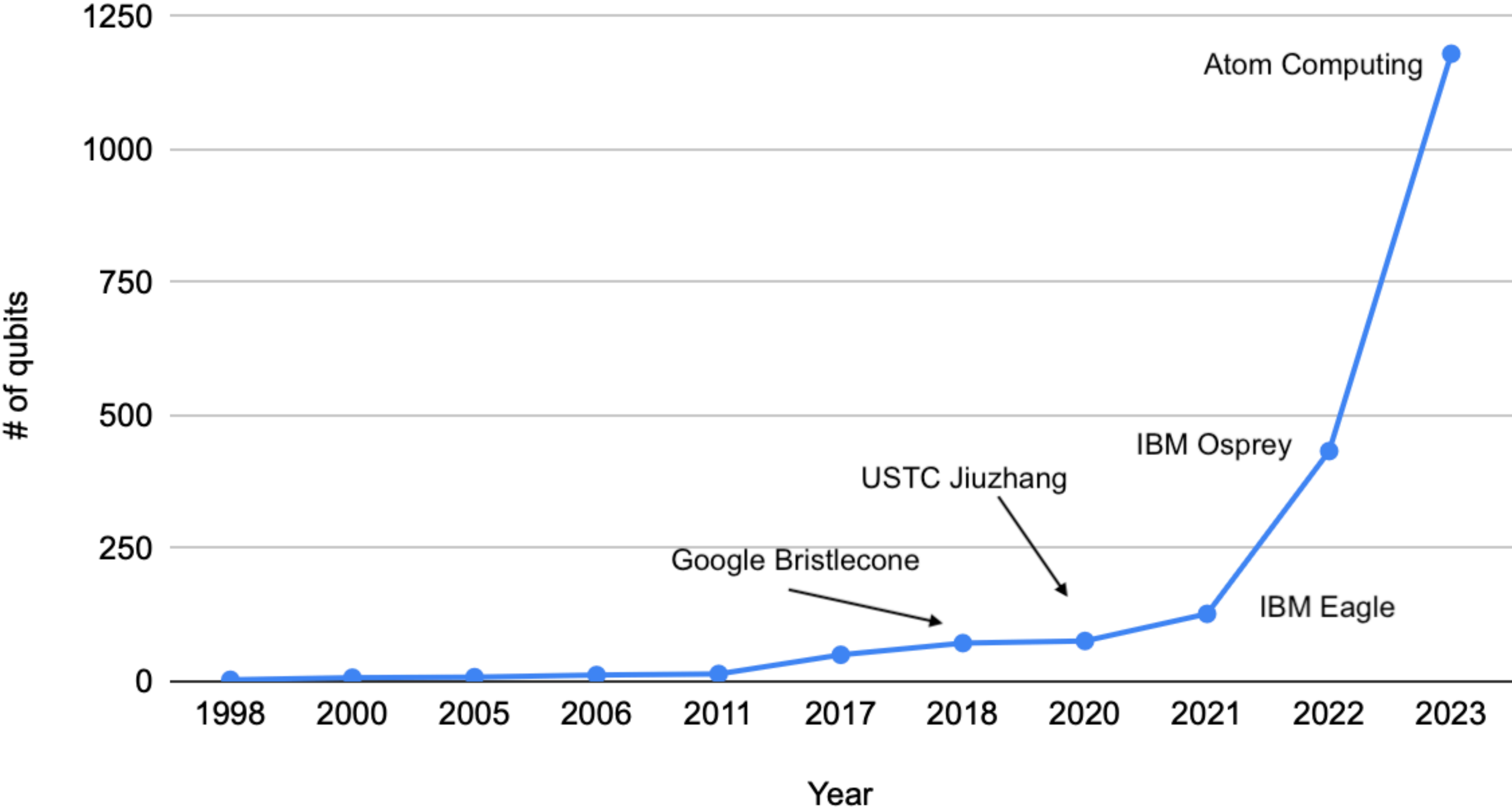
**Particle-Wave Duality
in a single slide**



Quantum Computing

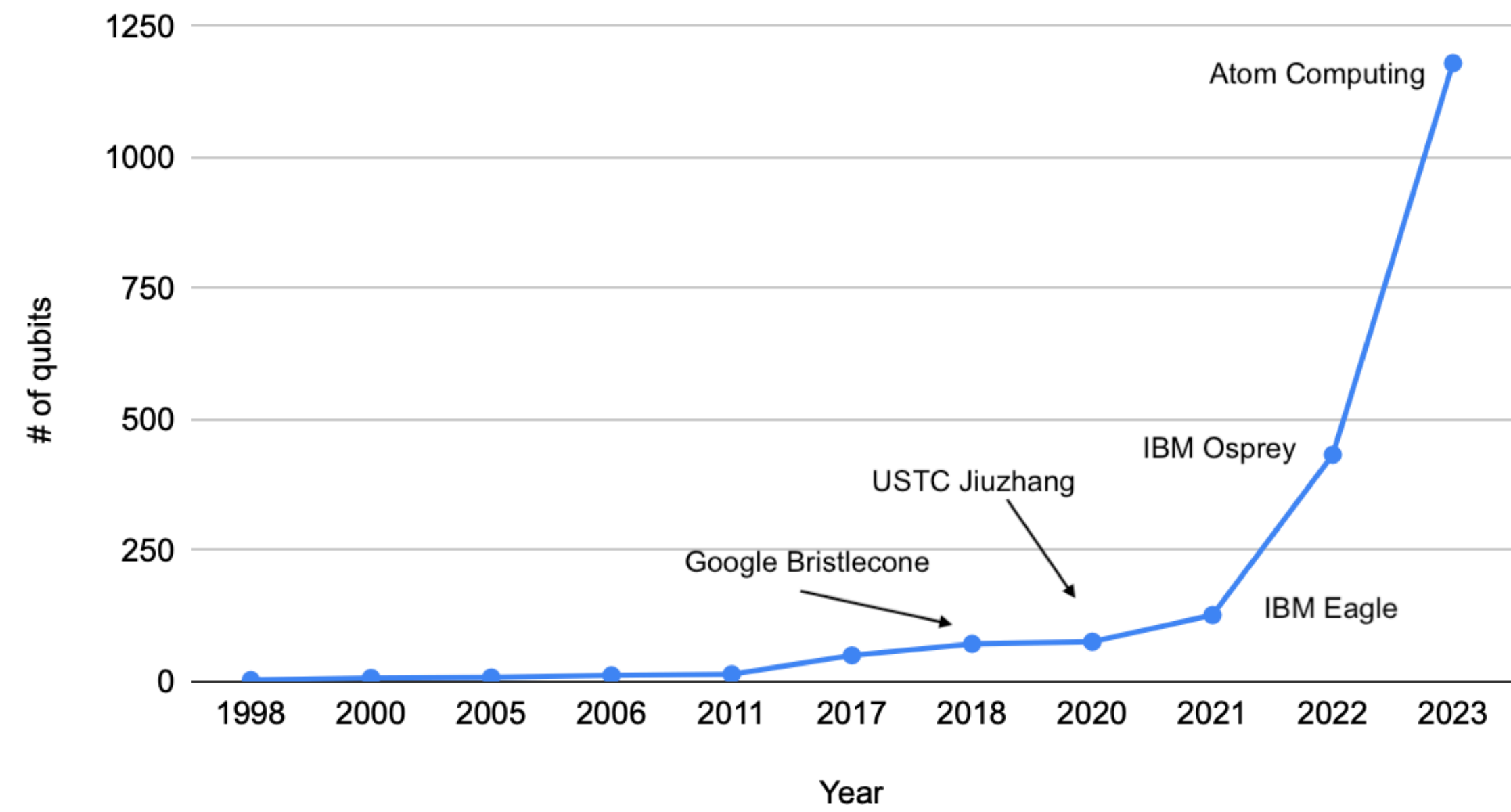


Quantum Computing Capability Over Time

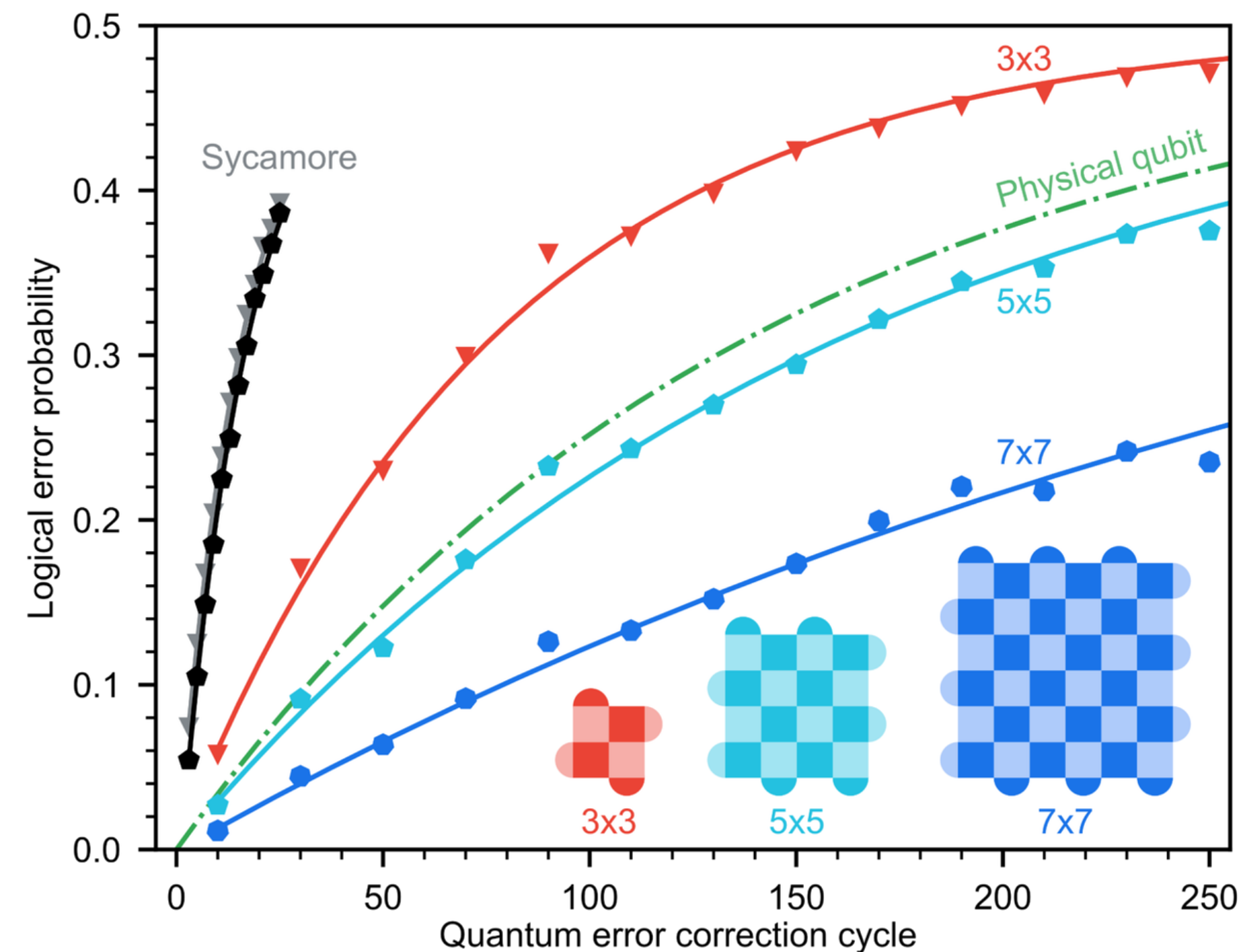




Quantum Computing Capability Over Time



- Microsoft's *Majorana 1* (8 topological qubits)
- Amazon's *Ocelot* chip (9 qubits)
- Alice & Bob Boson 4 (16 qubits)
- Google Willow (105 qubits)
- IBM Nighthawk (120 qubits)



Quantum Cryptography Threat Model

- **symmetric** key cryptography is subject to *Grover's algorithm*, which provides a *best case* quadratic speedup ($O(N) \rightarrow O(\sqrt{N})$), but doesn't lend itself to parallelization $\Rightarrow$ even 128 bit is considered safe
- **asymmetric** key cryptography (e.g., RSA, Diffie-Hellman incl. ECDH) is subject to *Shor's algorithm* for factorization; change of algorithms needed
- coherence time, quantum error-correction, and use of logical qubits all increase with time, leading to “**Q-Day**”, the day when CRQCs can break common cryptographic protocols





When does Q-Day arrive?

~~When does Q-Day arrive?~~

When do I need to have PQC deployed?

It doesn't matter if a CRQC doesn't manifest until after 2035 if all compliance requirements mandate a completed PQC migration by 2030.

NIST

NATIONAL INSTITUTE OF
STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

Mile

playbooks and inter
erprises: by 2033

ysical + PQC) add
all digital signat
ts, and contin
ne remaining lega

firmware upgrades are enabled by default.

low-risk use cases has been completed.

low-risk use cases has been completed as much as feasible.

- handle data that is highly sensitive, classified or has long-lived confidentiality requirements

- are likely to prove difficult or time consuming to update.

By end of 2030: Organisations should have completed their PQC transition.

disallowed and

ASD
AUSTRALIAN
SIGNALS
DIRECTORATE
ACSC
Australian
Cyber Security
Centre

Quantum Cryptography Threat Model

Got CRQC?

- quantum resistant: symmetric encryption (e.g., AES), integrity (e.g., SHA-2)
- quantum vulnerable:
 - asymmetric encryption / key exchange (e.g., RSA, ECDSA, ECDH)
 - signatures (e.g., RSA, ECDSA, ECDH)
- time horizon of a CRQC is (always?) 10+ years
- ...which gets us to Mosca's theorem

Mosca's Theorem



Mosca's Theorem

Shelf-life of secrets

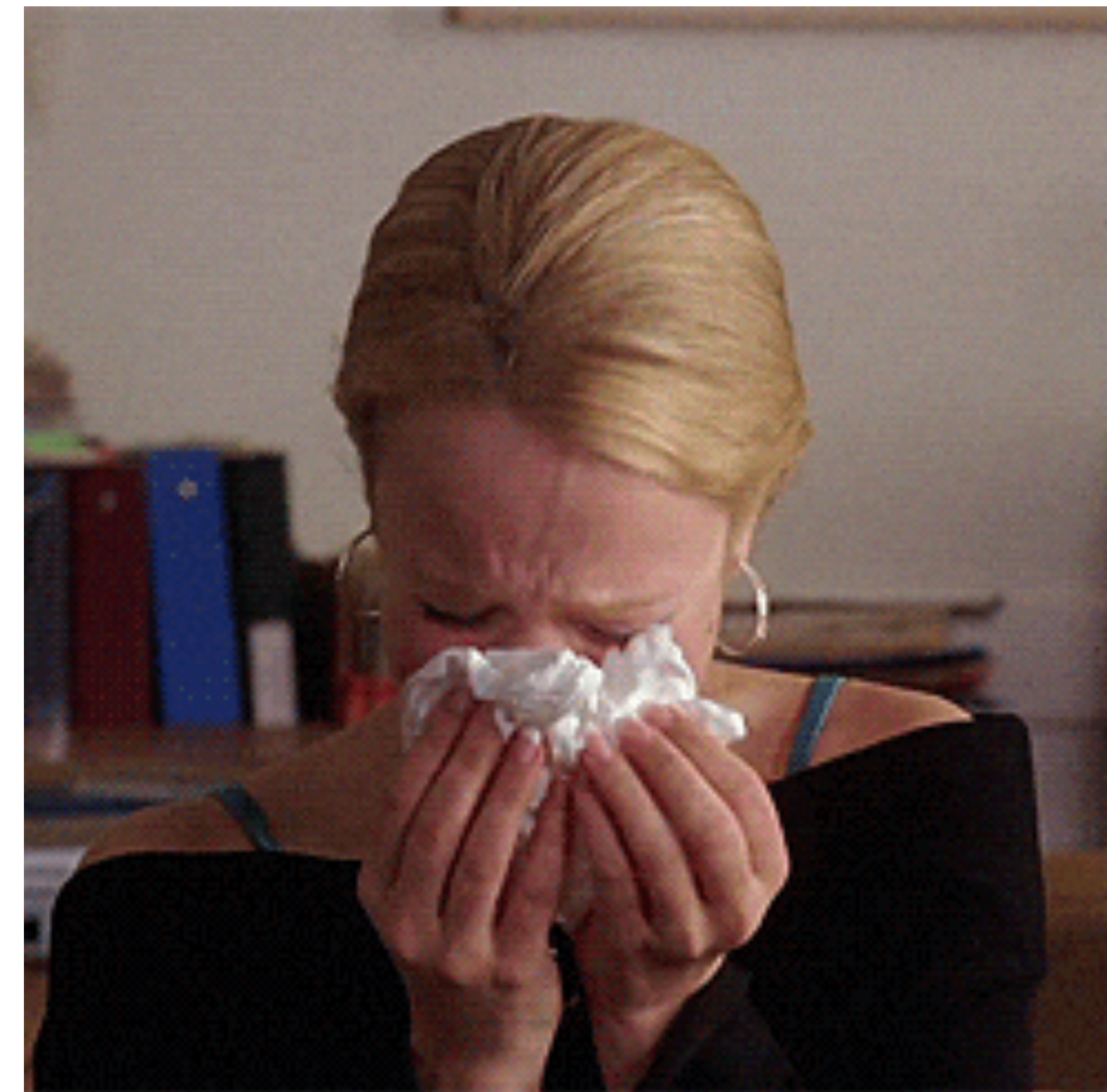
Time to Q-Day

If

$$X + Y > Q$$

Migration time

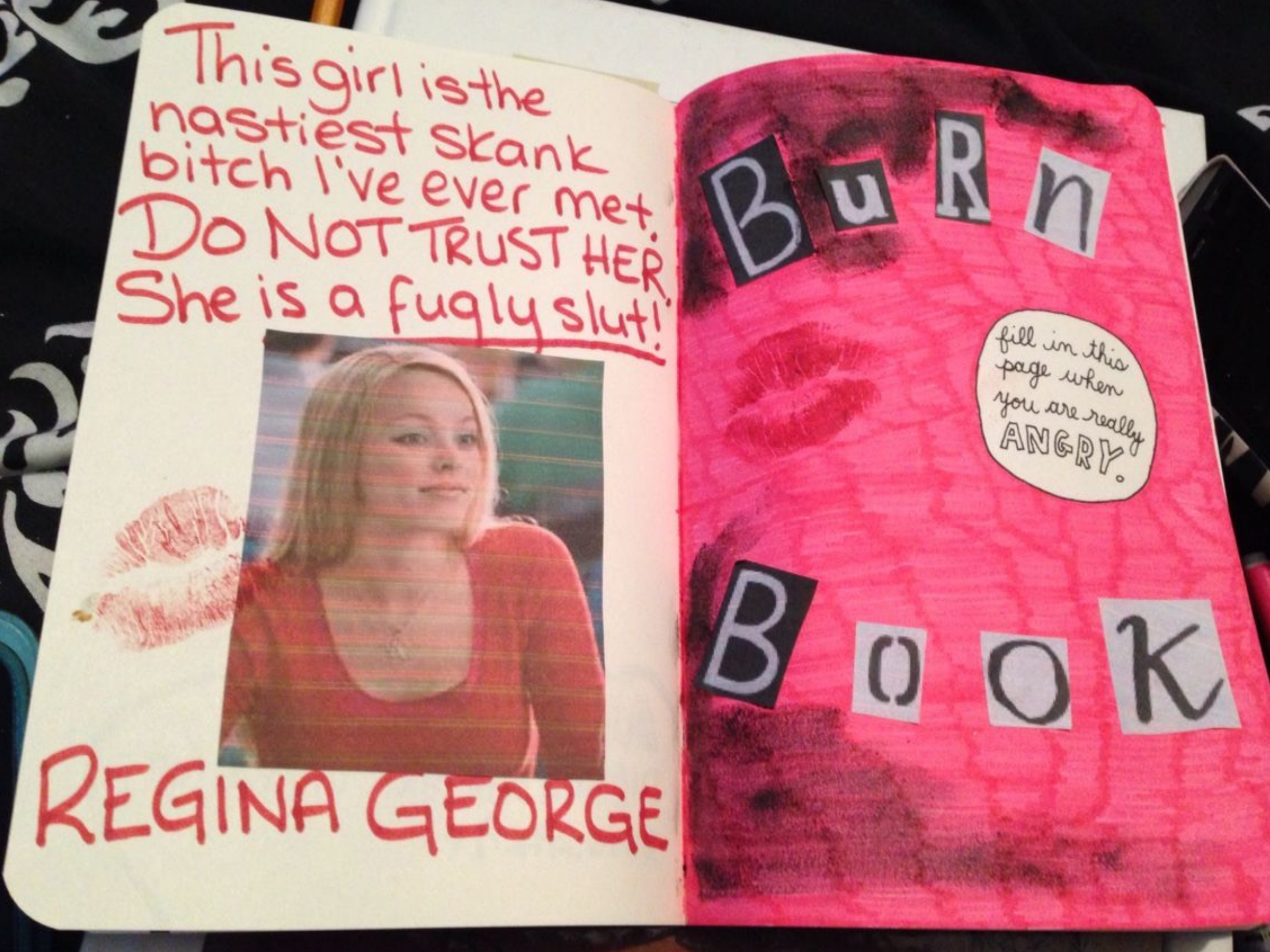
then



Quantum Cryptography Threat Model

Got CRQC?

- ~~quantum resistant: symmetric encryption (e.g., AES), integrity (e.g., SHA-2)~~
- quantum vulnerable:
 - asymmetric encryption / key exchange (e.g., RSA, ECDSA, ECDH)
 - signatures (e.g., RSA, ECDSA, ECDH)



Confidentiality



Authenticity

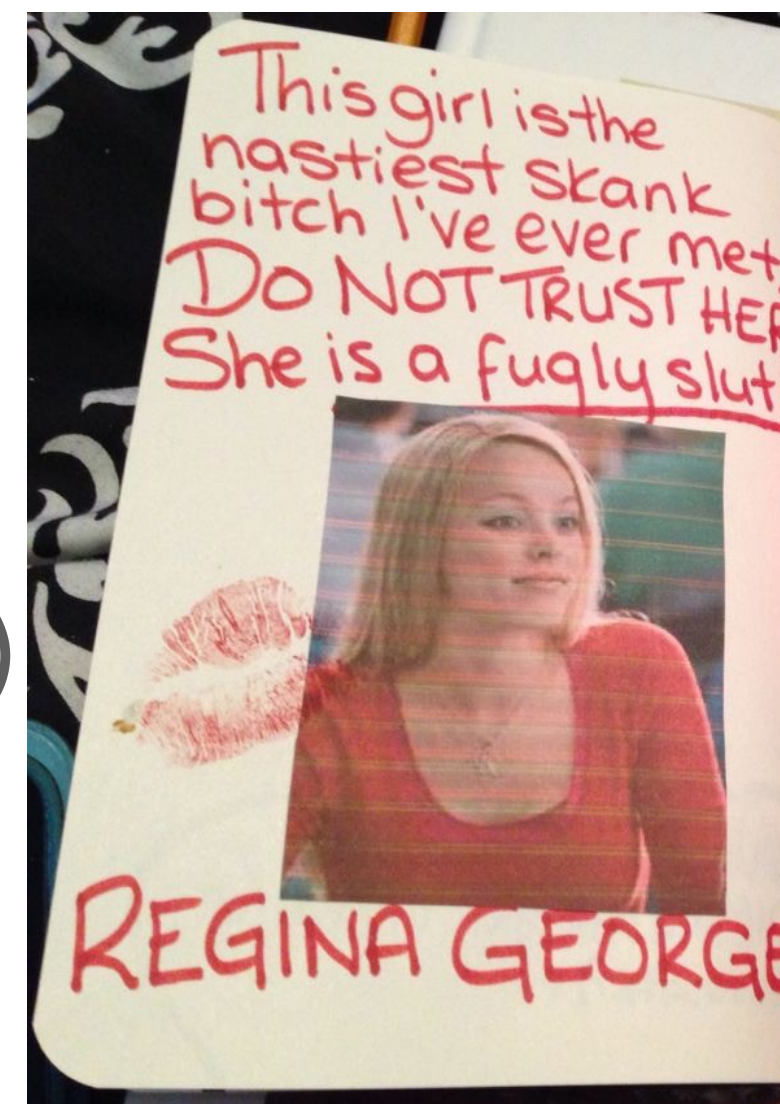


Post-Quantum Cryptography

NIST (eventually) to the rescue!

Post-Quantum Cryptography (PQC) refers to algorithms that are resistant to cryptanalytic attacks even using quantum computers.

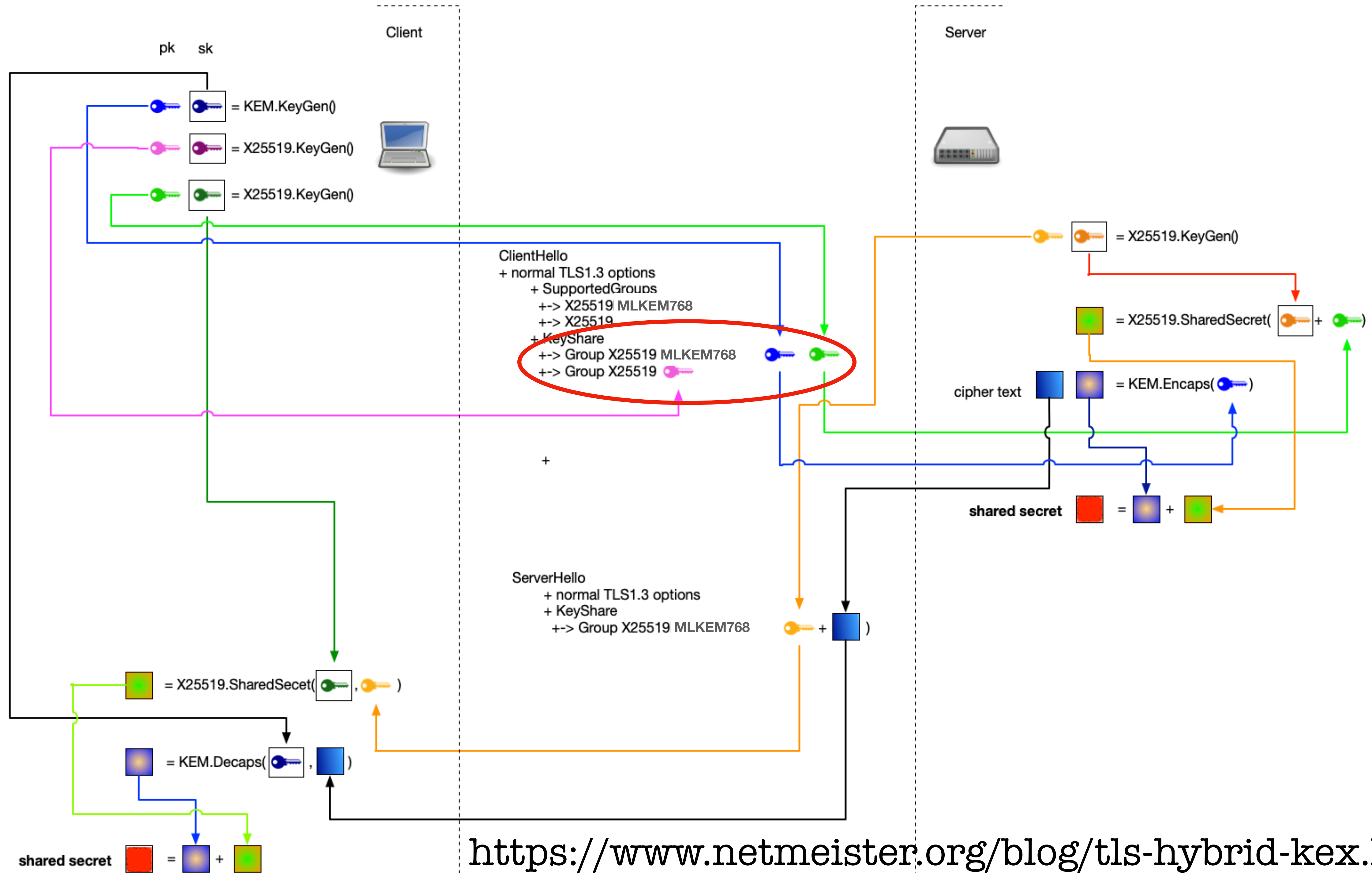
- Public-Key Encryption / Key Encapsulation Mechanism
 - FIPS-203: **ML-KEM** (Module-Lattice based Key Encapsulation Mechanism)
- Digital Signatures:
 - FIPS-204: **ML-DSA** (Module-Lattice based Digital Signature Algorithm)
 - FIPS-205: **SLH-DSA** (Stateless Hash-Based Digital Signature Algorithm)
 - FIPS 206: **FN-DSA** (Fast Fourier lattice-based compact signatures over NTRU) in 2026



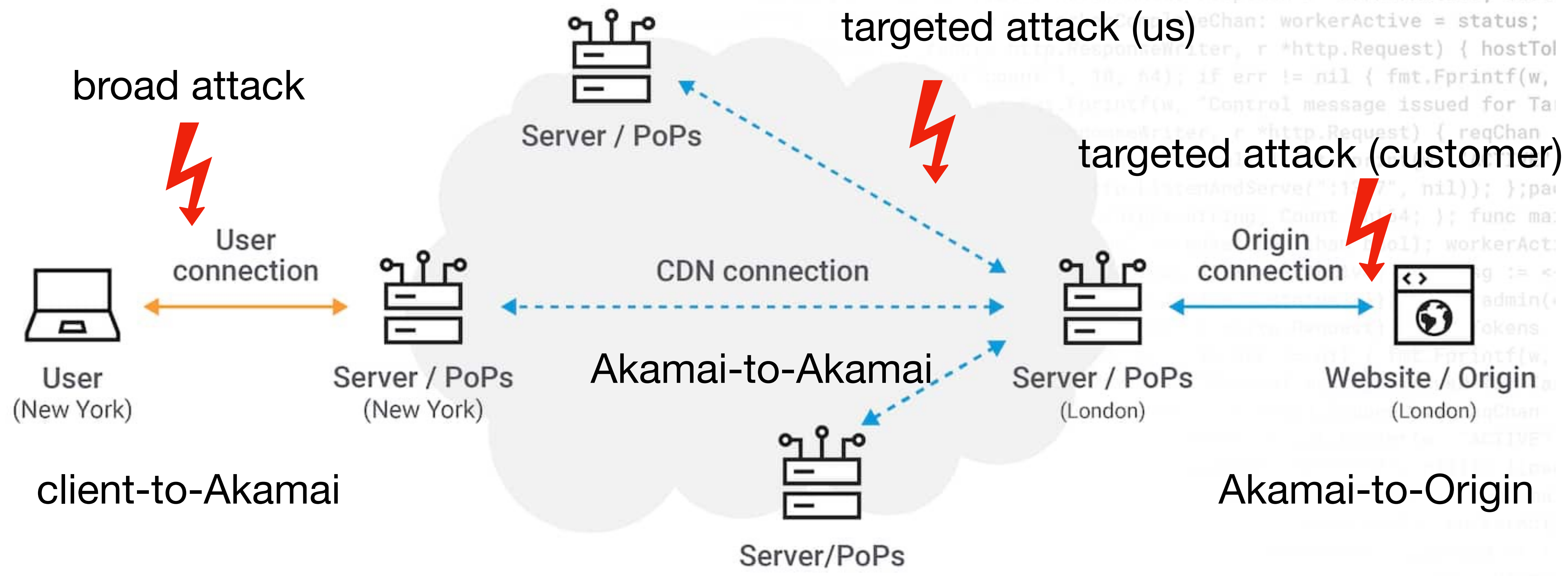
PQC Key Exchange

Meanwhile, at the IETF...

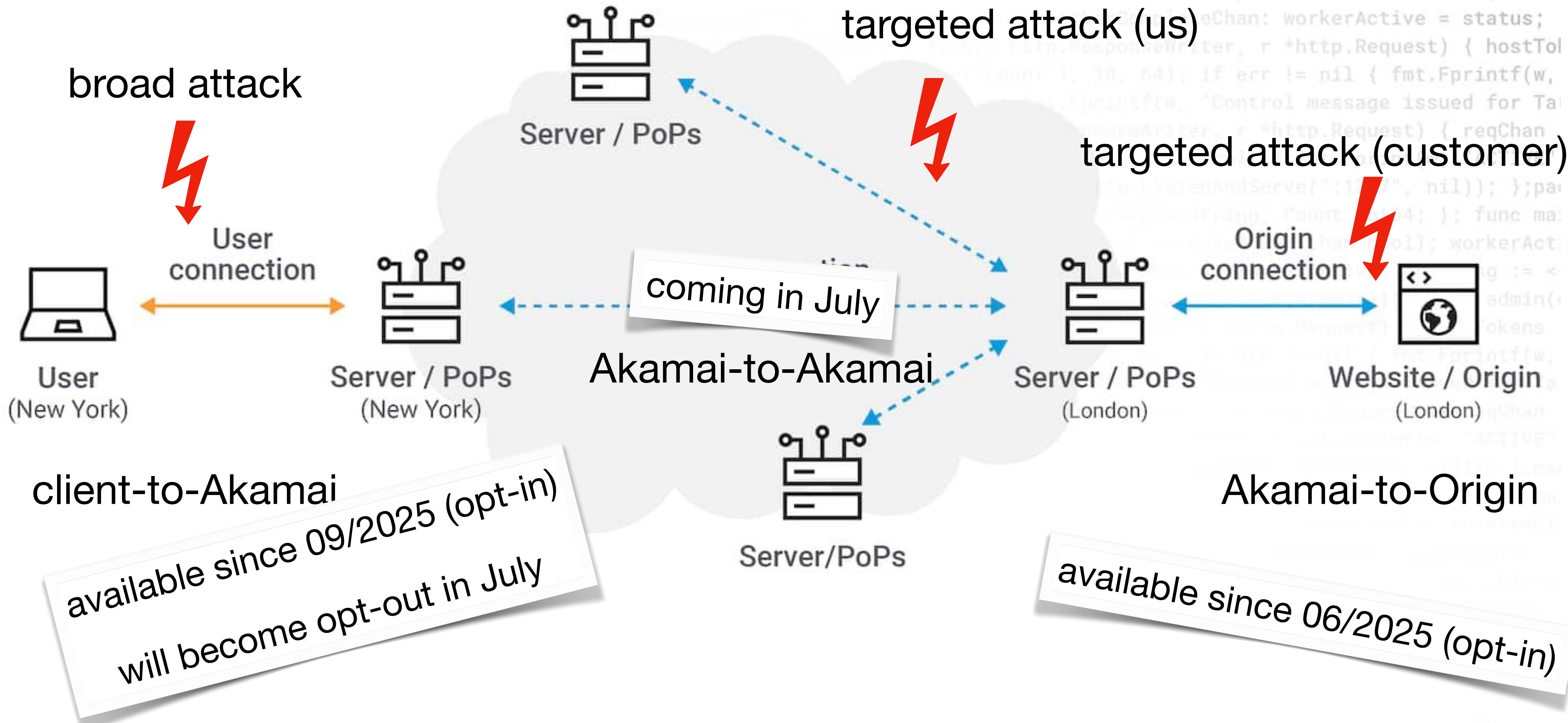
- PQC is new, implementations need to see the test of time (crypto is hard, mistakes are fatal)
- Solution: implement *hybrid* key exchange, combining PQC KEM with *classical* key exchange: <https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/>
- X25519MLKEM768 combines X25519 (RFC7748) Elliptic Curve with ML-KEM
- <https://datatracker.ietf.org/doc/draft-ietf-tls-ecdhe-mlkem/>



Akamai Threat Model

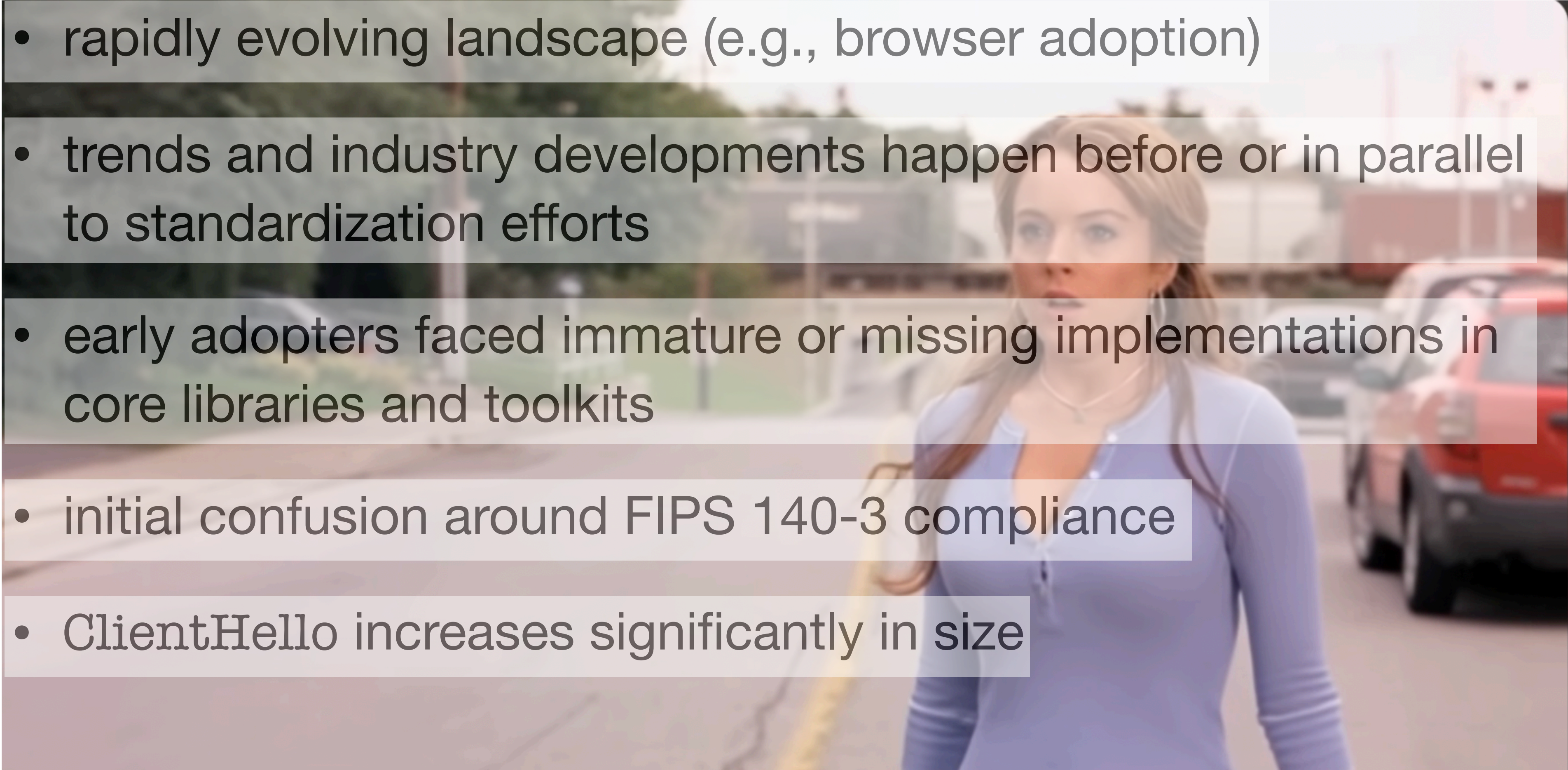


Akamai PQC Support



Challenges and Headaches

Things were moving fast...

- rapidly evolving landscape (e.g., browser adoption)
 - trends and industry developments happen before or in parallel to standardization efforts
 - early adopters faced immature or missing implementations in core libraries and toolkits
 - initial confusion around FIPS 140-3 compliance
 - ClientHello increases significantly in size
- 

Size matters

Key Exchange

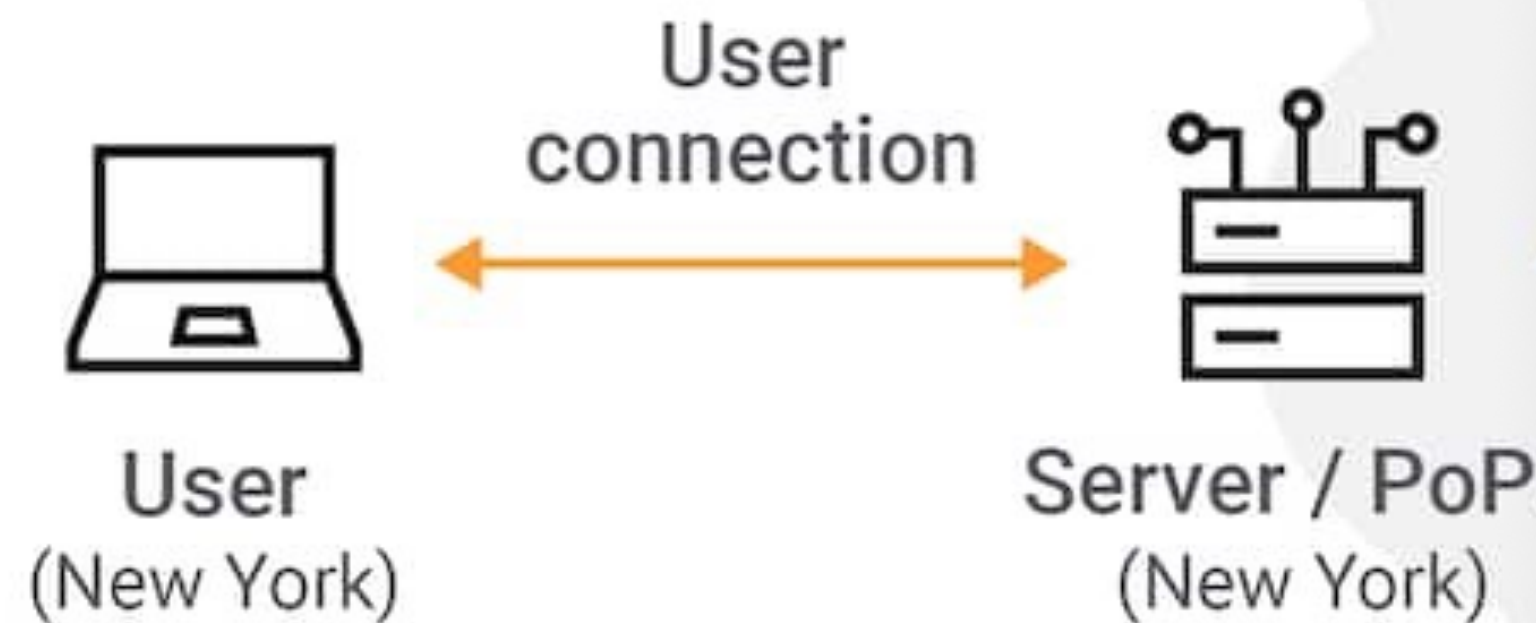
Algorithm	Public Key (bytes)
X25519	32
RSA-2048	256
ML-KEM768	1,184

Signature

Algorithm	Signature
Ed25519	32
RSA-2048	256
ML-DSA-44	2,420

- Hybrid Key Exchange ClientHello Key Shares increases from $1 \times 32 \rightarrow 1 \times 1,184 + 2 \times 32 = 1,248$ bytes
- Supporting multiple groups requires either:
 - out of band method to communicate preference (e.g., SVCB/HTTPS DNS records); or
 - increase ClientHello by adding additional (large) keys or possibly trigger HelloRetryRequest

Client-to-Akamai options



client-to-Akamai

```
Terminal — 80x24
[laptop$ </dev/null openssl s_client -curves '*X25519MLKEM768:*X25519' -connect w
ww.akamai.com:443 2>&1 | grep -C 9 "Negotiated TLS"
eCdQwHhtzJUyALK9zsHCCS5vsINNAC5GuADN++EZrCrAIXtFe+jVBV0FZ2s3XPru
iFJyAQgAQDD/1Q0+zQU=
-----END CERTIFICATE-----
subject=C=US, ST=Massachusetts, L=Cambridge, O=Akamai Technologies, Inc., CN=www
.akamai.com
issuer=C=US, O=DigiCert Inc, CN=DigiCert Global G2 TLS RSA SHA256 2020 CA1
---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: rsa_pss_rsae_sha256
Negotiated TLS1.3 group: X25519MLKEM768
---
SSL handshake has read 4719 bytes and written 1613 bytes
Verification: OK
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Protocol: TLSv1.3
Server public key is 2048 bit
This TLS version forbids renegotiation.
Compression: NONE
laptop$
```


out.pcap

Apply a display filter ... <%%/>

No.	Source	Destination	Protocol	Info
1	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	2600:141b:e800:3a::17ce:79d8	TCP	52854 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1440 WS
2	2600:141b:e800:3a::17ce:79d8	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	TCP	443 → 52854 [SYN, ACK, ECE] Seq=0 Ack=1 Win=64260 Len=0 MSS=1
3	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	2600:141b:e800:3a::17ce:79d8	TCP	52854 → 443 [ACK] Seq=1 Ack=1 Win=131904 Len=0 TSval=14193909
4	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	2600:141b:e800:3a::17ce:79d8	TCP	52854 → 443 [ACK] Seq=1 Ack=1 Win=131904 Len=1332 TSval=14193
5	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	2600:141b:e800:3a::17ce:79d8	TLSv1.3	Client Hello (SNI=www.akamai.com)
6	2600:141b:e800:3a::17ce:79d8	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	TCP	443 → 52854 [ACK] Seq=1 Ack=1534 Win=67200 Len=0 TSval=593412
7	2600:141b:e800:3a::17ce:79d8	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	TLSv1.3	Server Hello, Change Cipher Spec, Application Data
8	2600:141b:e800:3a::17ce:79d8	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	TCP	443 → 52854 [PSH, ACK] Seq=1429 Ack=1534 Win=67200 Len=1428 T
9	2600:141b:e800:3a::17ce:79d8	2600:4041:5ed1:aa00:cc66:fd71:a1ad:d749	TCP	443 → 52854 [PSH, ACK] Seq=2857 Ack=1534 Win=67200 Len=1240 T

- > Cipher Suites (30 suites)
- > Compression Methods Length: 1
- > Compression Methods (1 method)
- > Extensions Length: 1391
- > Extension: renegotiation_info (len=1)
- > Extension: server_name (len=19) name=www.akamai.com
- > Extension: ec_point_formats (len=2)
- ✓ Extension: supported_groups (len=6)
 - Type: supported_groups (10)
 - Length: 6
 - Supported Groups List Length: 4
 - ✓ Supported Groups (2 groups)
 - Supported Group: X25519MLKEM768 (0x11ec)
 - Supported Group: x25519 (0x001d)
- > Extension: session_ticket (len=0)
- > Extension: encrypt_then_mac (len=0)
- > Extension: extended_master_secret (len=0)
- > Extension: signature_algorithms (len=54)
- > Extension: supported_versions (len=5) TLS 1.3, TLS 1.2
- > Extension: psk_key_exchange_modes (len=2)
- ✓ Extension: key_share (len=1258) X25519MLKEM768, x25519
 - Type: key_share (51)
 - Length: 1258
 - ✓ Key Share extension
 - Client Key Share Length: 1256
 - ✓ Key Share Entry: Group: X25519MLKEM768, Key Exchange length: 1216
 - Group: X25519MLKEM768 (4588)
 - Key Exchange Length: 1216
 - Key Exchange [...]: 54d3ac185b92f0094292eb8fdc07b26398ccc08b176f22ab042bc490a4bf1e41205b648ebc67666dc4241118cbc69c5ad6b7bad471a22b508cd376ae31722e7...
 - > Key Share Entry: Group: x25519, Key Exchange length: 32

- [JA4: t13d301100_1d37bd780c83_8e6e362c5eac]
- [JA4_r [...]: t13d301100_002f,0033,0035,0039,003c,003d,0067,006b,009c,009d,009e,009f,1301,1302,1303,c009,c00a,c013,c014,c023,c024,c027,c028,c02b,c02c,c02f,c...
- [JA3 Fullstring: 771,4866-4867-4865-49196-49200-159-52393-52392-52394-49195-49199-158-49188-49192-107-49187-49191-103-49162-49172-57-49161-49171-51-157-15...
- [JA3: c63f7f3dc56f593328fb18040c0152d1]

Group (tls.handshake.extensions_key_share_group), 2 bytes

Packets: 22

Profile: Default

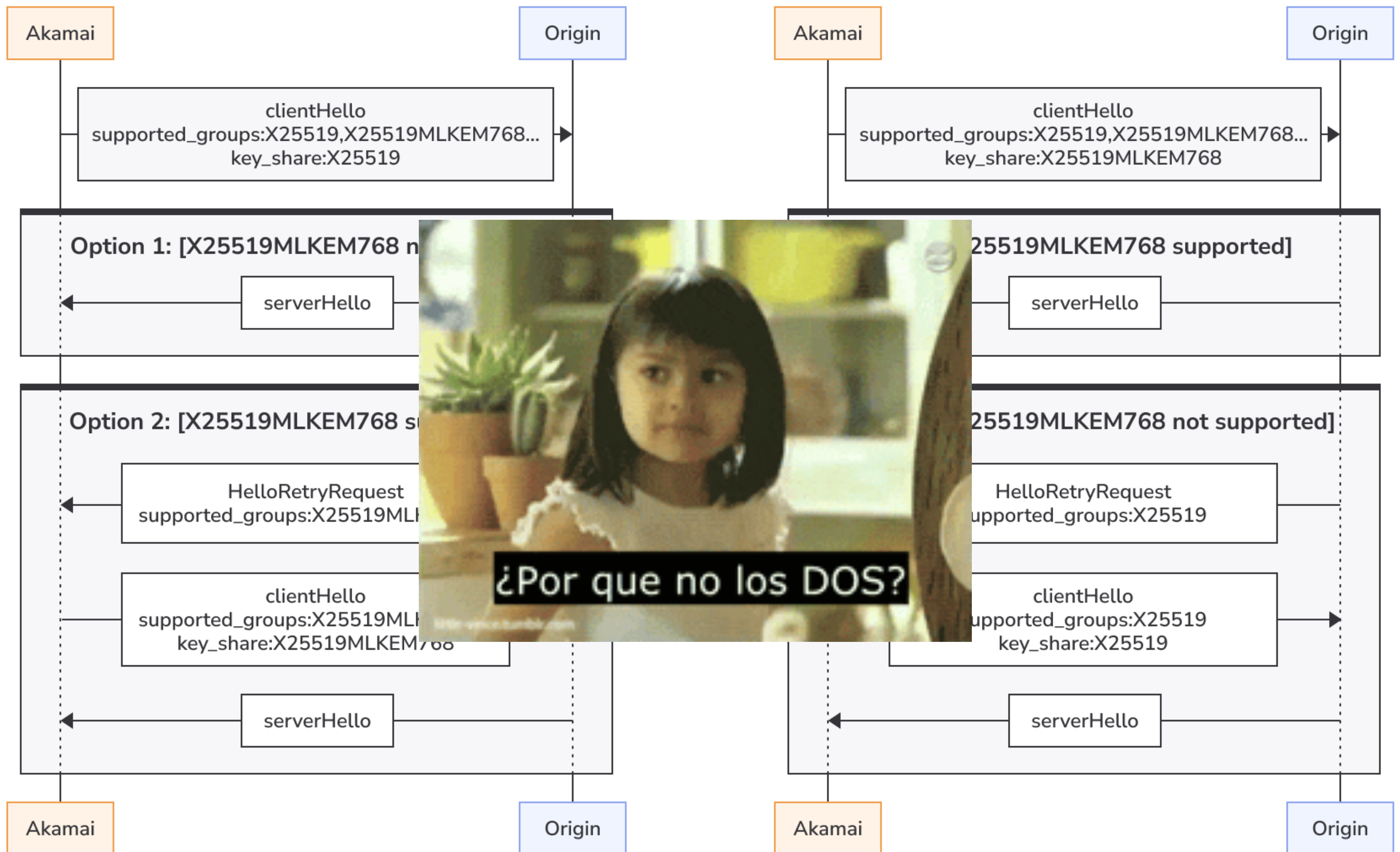
```
et { Target string; Co  
ake(chan chan bool);  
<- workerActive: case
```

```
:*X25519'-connect w
```

XPr

logies, Inc., CN=www

HA256 2020 CA1



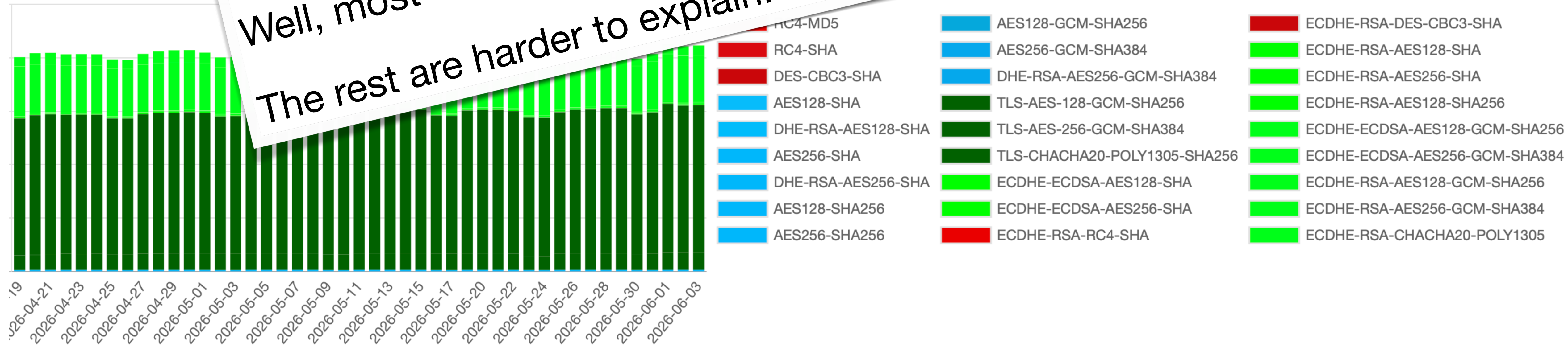
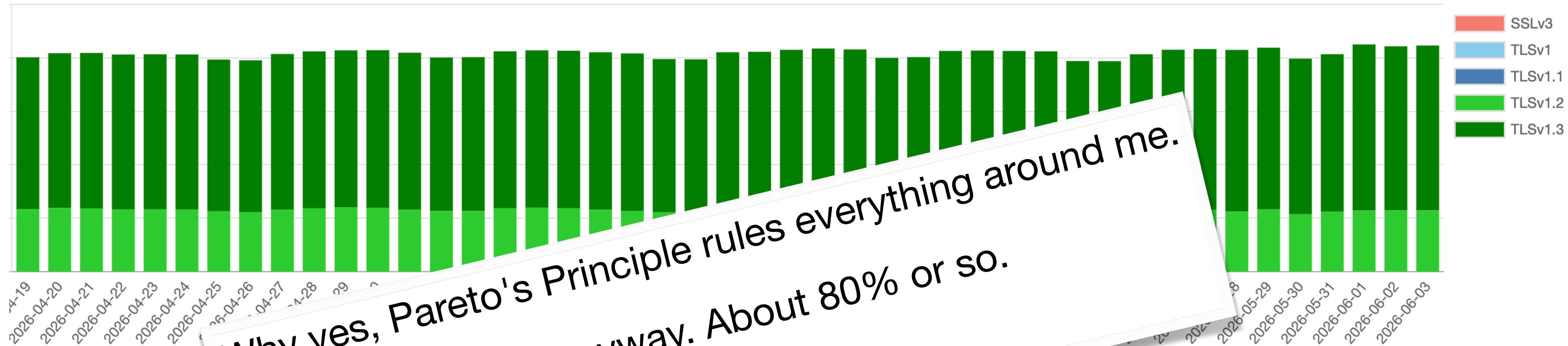
Customers do the darndest things

(We love our customers, tho.)

- Origins run on ~about every conceivable stack, including other CDNs, cloud providers, proprietary and homegrown servers, third-party services and middleboxes, supported and unsupported Open Source projects, ...
- Some vendors / customers will only support what is written in stone an RFC
- Customers often require specific exceptions to RFCs
- Protocol ossification has long been a problem (despite GREASE)

Akamai-scale means even very small percentages still mean **lots** of traffic.

Protocol data by day:



Why yes, Pareto's Principle rules everything around me.
Well, most things, anyway. About 80% or so.
The rest are harder to explain.

Quantum Cryptography Threat Model

Got CRQC?

- ~~quantum resistant: symmetric encryption (e.g., AES), integrity (e.g., SHA-2)~~
- quantum vulnerable:

$$X + Y > Q$$

 - asymmetric encryption / key exchange (e.g., RSA, ECDSA, ECDH)
 - signatures (e.g., RSA, ECDSA, ECDH)
- ~~time horizon of a CRQC is (always?) 10+ years~~
- what if $Q \cong \text{now}$?

Securing Elliptic Curve Cryptocurrencies against Quantum Vulnerabilities: Resource Estimates and Mitigations

Ryan Babbush,^{1,*} Adam Zalcman,^{1,†} Craig Gidney,^{1,‡} Michael Broughton,¹
Tanuj Khattar,¹ Hartmut Neven,¹ Thiago Bergamaschi,^{1,2} Justin Drake,³ and Dan Boneh⁴

¹*Google Quantum AI, Santa Barbara, CA 93111, United States*

²*Department of Computer Science, University of California Berkeley, Berkeley, CA 94720, United States*

³*Ethereum Foundation, Zeughausgasse 7a, 6300 Zug, Switzerland*

⁴*Department of Computer Science, Stanford University, Stanford, CA 94305, United States*

(Dated: April 17, 2026)

April 15th, 2026

The expected emergence of cryptographically relevant quantum computers (CRQCs) will represent a singular discontinuity in the history of digital security, with wide ranging impacts. This whitepaper seeks to elucidate specific implications that the capabilities of developing quantum architectures have on blockchain vulnerabilities and potential mitigation strategies. First, we provide new resource estimates for breaking the 256-bit Elliptic Curve Discrete Logarithm Problem over the secp256k1 curve, the core of modern blockchain cryptography. We demonstrate that Shor's algorithm for this problem can execute with either ≤ 1200 logical qubits and ≤ 90 million Toffoli gates or ≤ 1450 logical qubits and ≤ 70 million Toffoli gates. In the interest of responsible disclosure, we use a zero-knowledge proof to validate these results without disclosing attack vectors. On superconducting architectures with 10^{-3} physical error rates and planar connectivity, those circuits can execute in minutes using fewer than half a million physical qubits. We introduce a critical distinction between "fast-clock" (such as superconducting and photonic) and "slow-clock" (such as neutral atom and ion trap) architectures. Our analysis reveals that the first fast-clock CRQCs would enable "on-spend" attacks on public mempool transactions of some cryptocurrencies. We survey major cryptocurrency vulnerabilities through this lens, identifying systemic risks associated with advanced features in some blockchains such as smart contracts, Proof-of-Stake consensus, and Data Availability Sampling mechanism, as well as the enduring concern of "abandoned" assets. We argue that technical solutions would benefit from accompanying public policy and discuss various frameworks of "digital salvage" to regulate the recovery or destruction of dormant assets while preventing adversarial seizure. We also discuss implications for other digital assets and tokenization as well as challenges and successful examples of the ongoing transition to Post-Quantum Cryptography (PQC). Finally, we urge all vulnerable cryptocurrency communities to join the migration to PQC without delay.

"nearly a 20 fold reduction
over prior estimates"

as 10,000 reconfigurable atomic qubits

ng¹, Lewis R. B. Picard¹, Harry Levine^{1,3},
-Yuan Huang^{1,2}, Dolev Bluvstein^{1,2,§}
alifornia 91125, USA
Pasadena, California 91125, USA
fornia, Berkeley, California 94720, USA
ic.com, §dbluvstein@oratomic.com
tributed equally
n 31, 2026)

form computational tasks beyond the reach of
r's algorithm for integer factorization and dis-
ortance and practical relevance to cryptography.
for correction, optimized resource estimates for
thm require millions of physical qubits. Here, by
rrecting codes, efficient logical instruction sets,
can be executed at cryptographically relevant
qubits. Increasing the number of physical qubits
elism; under plausible assumptions, the runtime
ould be just a few days for a system with 26,000
-2048 integers is one to two orders of magnitude
onstrated universal fault-tolerant operations be-
arrays of hundreds of qubits, and trapping arrays
ough substantial engineering challenges remain,
ately designed neutral-atom architecture could
ly relevant scales. More broadly, these results
tolerant quantum computing with wide-ranging

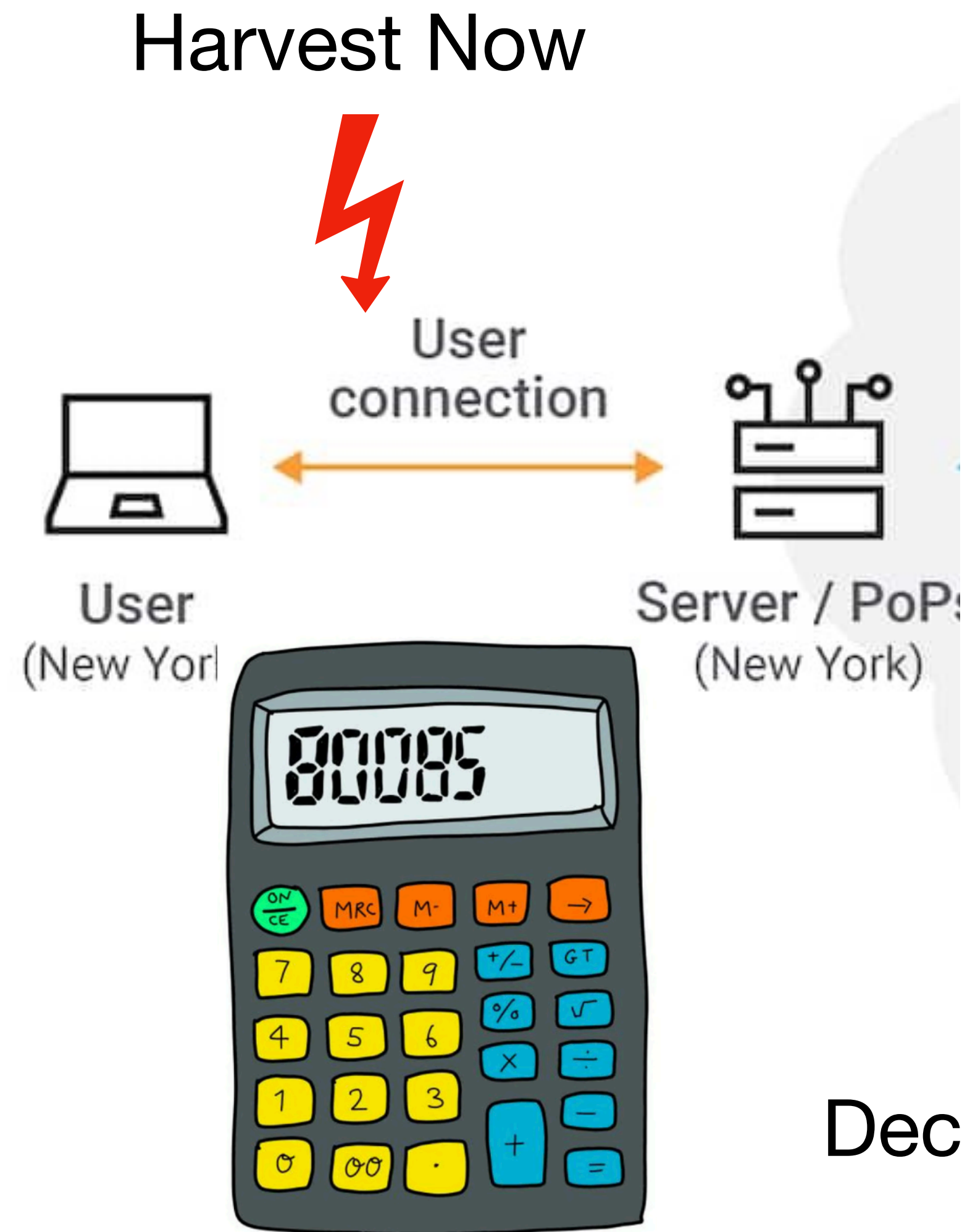
March 30th, 2026

Mosca's Theorem re-evaluated

$$\lim_{t \rightarrow Y} P(Q) = 1$$



Threat Model for **Y** >> now



1. Opportunistically Hoover up all data
2. Painstakingly decrypt each individual TLS key exchange

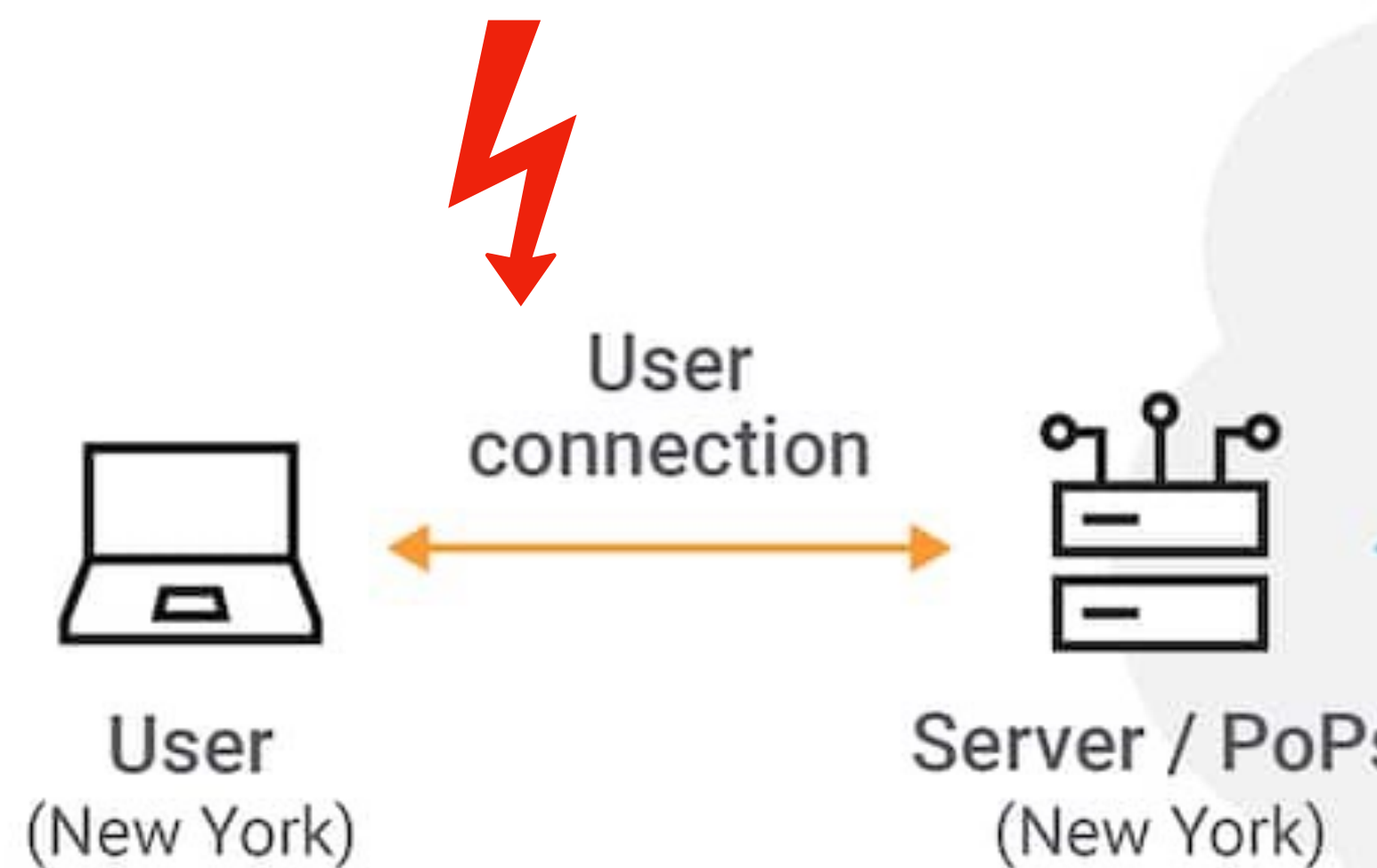
Only practical with time. (And still expensive.)

Decrypt Later

Threat Model for **Y** = now

```
type ControlMessage struct { Target string; Co
```

MitM your target



1. Grab a non-PQC cert
2. Use CRQC to factor RSA / solve discrete log
3. Mint your fraudulent cert
4. MitM your target
5. Enjoy access to all connections in real time

While you're at it, also pwn:

electronic signatures, code signing certs, trusted boot, mTLS auth, ...

Mosca's Theorem re-evaluated

If $\lim_{t \rightarrow Y} P(Q) = 1$ then



Your threat model shifts:

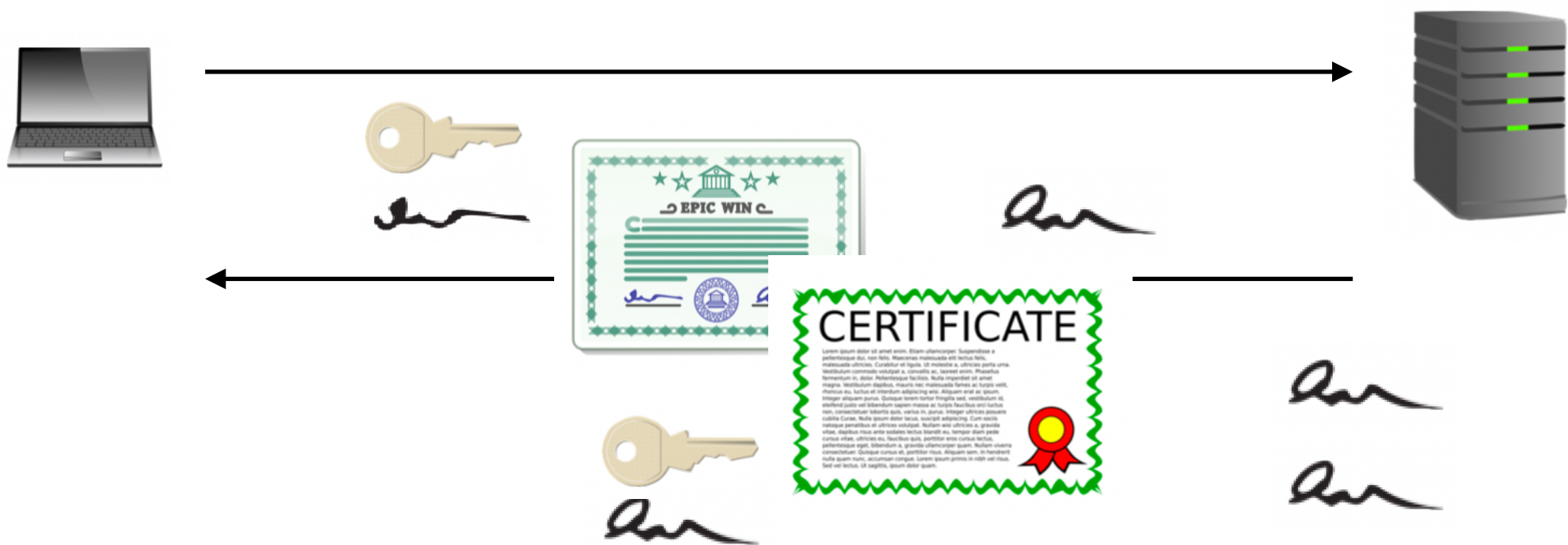
Threat of impersonation > threat of HNDL.

Size matters

$ECDSA\ P-256: 64 + 3 * 64 + 96 + 2 * 96 = 544$

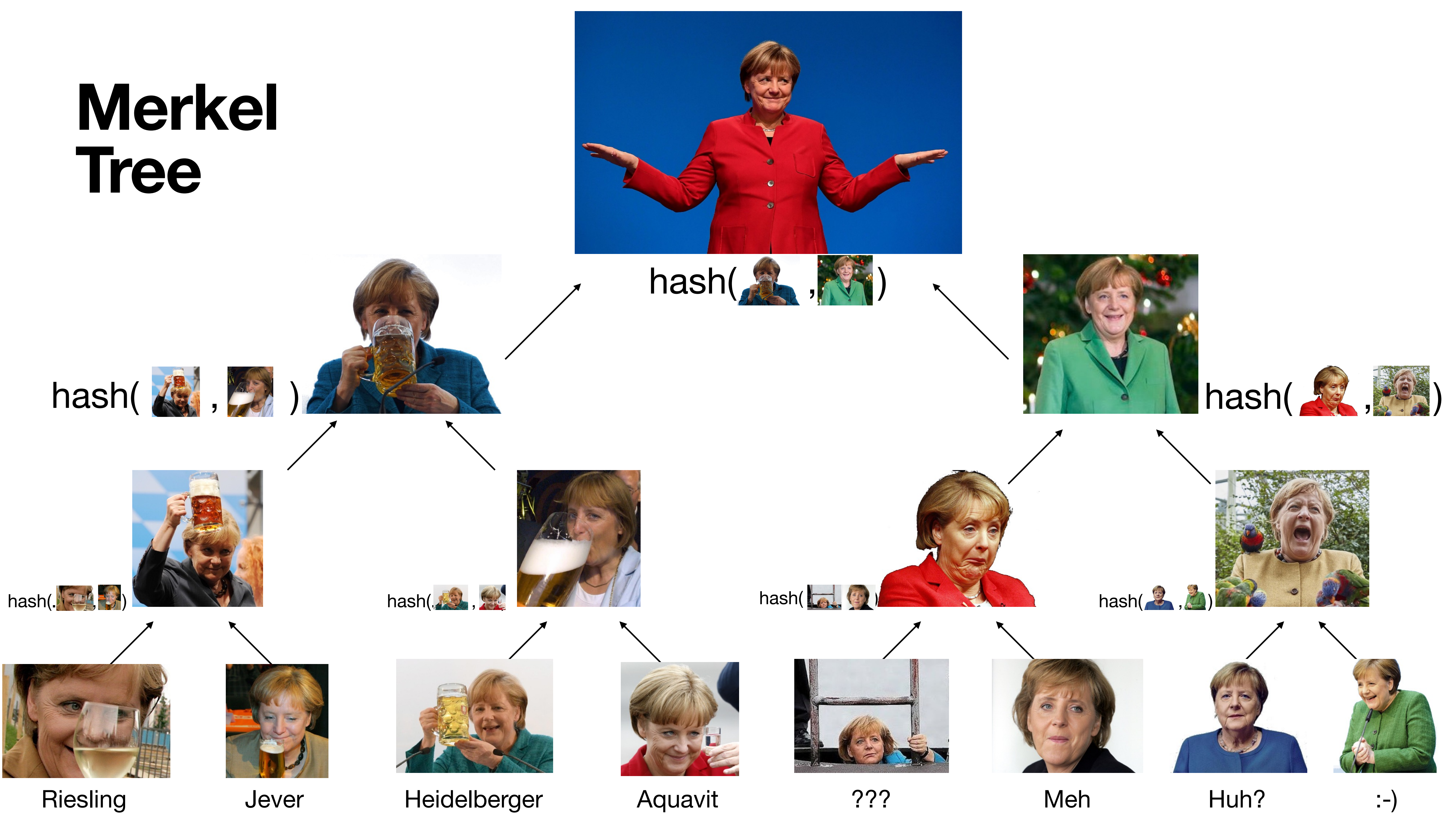
$ML\text{-}DSA\text{-}44: 2 * 1,312 + 5 * 2,420 = 14,724$

Algorithm	Public Key	Signature (raw)
ECDSA P-256	64	64
ECDSA P-384	96	96
RSA-2048	256	256
ML-DSA-44 (NIST Level 1 / 128-bit)	1,312	2,420
ML-DSA-65 (NIST Level 3 / 192-bit)	1,952	3,309
ML-DSA-87 (NIST Level 5 / 256-bit)	2,592	4,627



- Leaf cert:
 - pubkey + signature + SCTs
- Intermediate cert:
 - pubkey + signature
- transcript hash signature

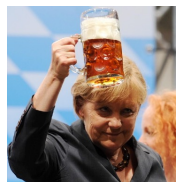
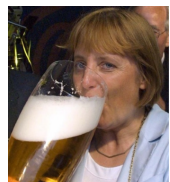
Merkel Tree



Merkel Tree



hash(, )

hash(, )



hash(, )



Riesling

Jever

Heidelberger

Aquavit

???

Meh

Huh?

: -)

Merkle Tree Certificate

Standalone certificates

ECDSA P-256: $64 + 3 * 64 + 96 + 2 * 96 = 544$

ML-DSA-44: $2 * 1,312 + 5 * 2,420 = 14,724$

Goal: avoid having to send multiple large signatures

To issue a new MTC, the CA...

- ...adds cert info to its logs
- ...constructs and signs checkpoint
- ...determines and signs required subtrees
- ...collects 2+ cosignatures (similar to collecting SCTs from CT logs)

1 TLS transcript hash signature: 2,420 bytes +

1 public key: 1,312 bytes +

1 inclusion proof: 384 bytes +

2(+) cosignatures: $2 * 2,420 =$

Total: 8,956 bytes

Merkle Tree Certificate

Landmark-relative certificates

- Instead of signing each certificate individually, certificates are signed in batches, entering information into a Merkle tree
- Certificates include an inclusion proof referred to as a “landmark”
- Landmarks from trusted cosigners (trust anchors) are signed periodically; client signals availability of landmark to server

$$\text{ECDSA P-256: } 64 + 3 * 64 + 96 + 2 * 96 = 544$$

$$\text{ML-DSA-44: } 2 * 1,312 + 5 * 2,420 = 14,724$$

1 TLS transcript hash signature: 2,420 bytes +

1 public key: 1,312 bytes +

1 inclusion proof: 384 bytes +

2(+) cosignatures: $2 * 2,420 =$

Total: 8,956 bytes

1 TLS transcript hash signature: 2,420 bytes +

1 public key: 1,312 bytes +

1 inclusion proof: 736 bytes =

Total: 4,468 bytes

Merkle Tree Certificates

Considerations

- Issuance of landmark-relevant certificates incur some delay.
- Client needs to know which landmarks to signal to which server.

We'll stuff it into the DNS.
Duh.



Merkle Tree Certificates

Considerations

- Issuance of landmark-relevant certificates incur some delay.
- Client needs to know which landmarks to signal to which server.
- Server needs to have *both* standalone and landmark-relevant certs ready.
- Client needs to update landmarks - frequently!



Are we (further) bifurcating the Web PKI (and browsers specifically) from other TLS use cases?

Merkle Tree Certificates

Considerations

- Issuance of landmark-relevant certificate
- Client needs to know which landmarks to
- Server needs to have *both* standalone and
- Client needs to update landmarks - frequently:

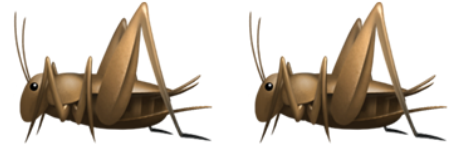


Are we (further) bifurcating the Web PKI (and browsers specifically) from other TLS use cases?

PQC Adoption

Certificates

The logo for MTCS (Multi-Trust Certificate System) is displayed in a white box with a shadow. It consists of the letters 'MTCS' in a bold, black, sans-serif font.

- Google, Cloudflare collaborating on feasibility study
- Let's Encrypt targets 2027 for production-ready support
- Google targets Q3 2027 for Chrome Quantum-resistant Root Store (CQRS)
- CA/B Forum... 

The logo for ML-DSA (Multivariate Polynomial Digital Signature Algorithm) is displayed in a white box with a shadow. It consists of the text 'ML-DSA' in a bold, black, sans-serif font.

- Google Chrome 150 will support ML-DSA for private PKI
- Microsoft signaled plans to support ML-DSA-87
- CA/B Forum...
 - Web-PKI Baseline Requirements still don't allow ML-DSA
 - SMIME BRs allow ML-DSA, ML-KEM
- Financial industry shared private ASC X9 PKI (run by DigiCert) offers ML-DSA roots

PQC Adoption

Certificates

MTCS

- Google, Cloudflare collaborating on feasibility study

- Google Chrome for private
- Microsoft s DSA-87



So private and public PKI will work fundamentally differently now?

(CQRS)

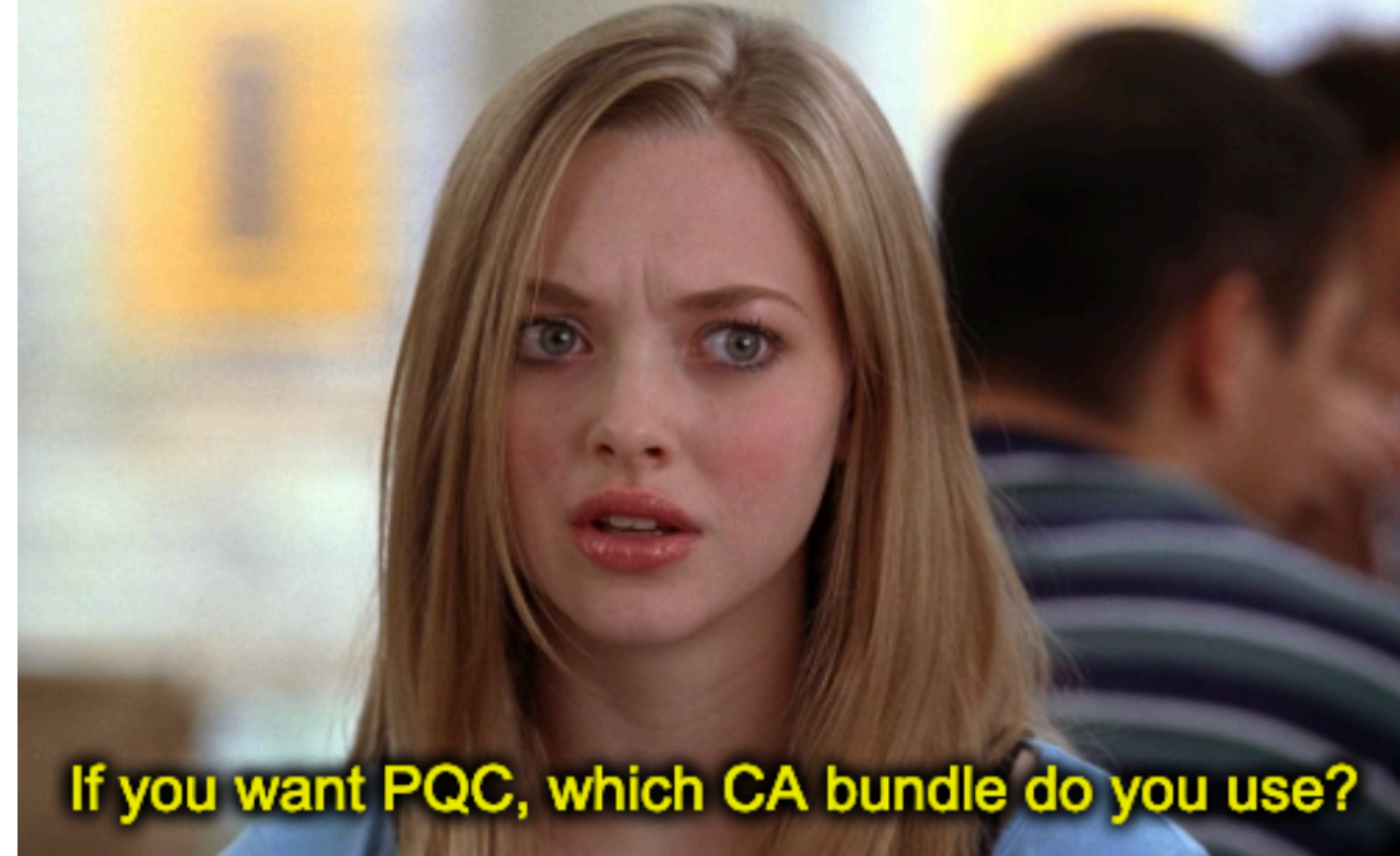
Baseline Requirements still
w/ ML-DSA

- SMIME BRs allow ML-DSA, ML-KEM
- Financial industry shared private ASC X9 PKI (run by DigiCert) offers ML-DSA roots

Web- vs private PKI

Just some possible locations:

- built-in (e.g., browsers)
- `/etc/openssl/certs/ca-bundle.crt`
- `/etc/openssl/cert.pem`
- `/etc/ssl/certs/ca-bundle.crt`
- `/etc/ssl/cert.pem`
- `/etc/pki/tls/cert.pem`
- `/etc/pki/tls/certs/ca-bundle.crt`
- `/etc/pki/ca-trust/extracted/pem/tls-ca-bundle.pem`
- any of the above, but under `/usr/local` or `/opt` somewhere
- whatever a certain environment variable (e.g., `CURL_CA_BUNDLE`, `REQUESTS_CA_BUNDLE`, `SSL_CERT_FILE`, ...) points to
- whatever a certain inconsistent command-line option specifies (e.g., `--cacert` (`curl(1)`), `--ca-certificate` (`wget(1)`), `-CAfile` (`s_client(1)`))
- ...



If you want PQC, which CA bundle do you use?



OMG Karen, you can't just ask someone what CA bundle they use.

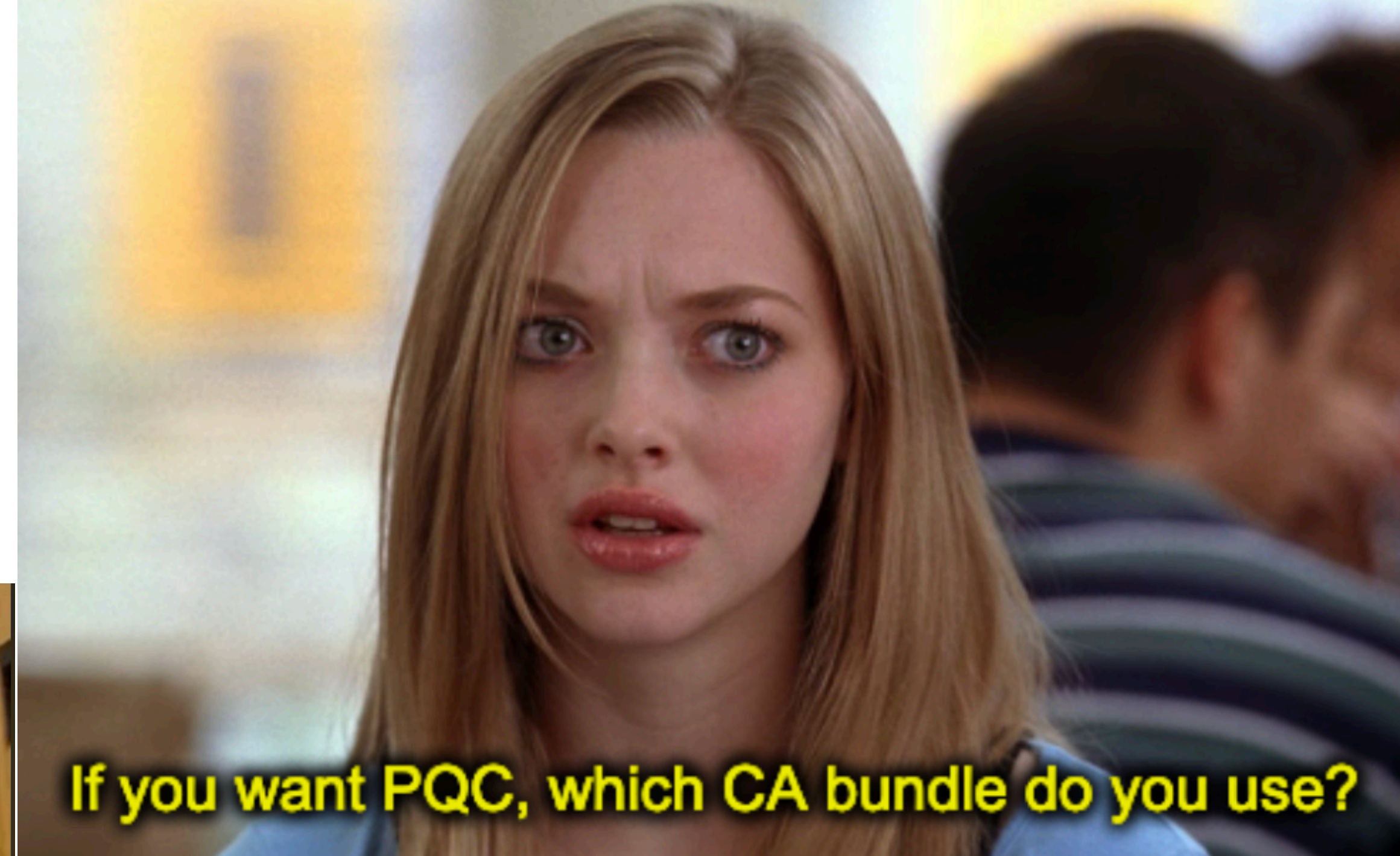
Web Psych! Everybody uses **KL**
Just so `curl -k` anyway.

- built-in (e.g., browsers)



(e.g., `--cacert (curl(1))`, `--ca-certificate (wget(1))`, `-CAfile (s_client(1))`)

- ...



If you want PQC, which CA bundle do you use?



OMG Karen, you can't just ask someone
what CA bundle they use.

Lessons

- Customers require clarity:
 - Client-to-Akamai and Akamai-to-Origin are independent
 - Key Exchange and certs are independent
 - You need TLS 1.3
 - You need a reasonably up-to-date serving stack

Lessons

You need to deliver right now but also not move too fast.

Customers:



We need PQC **ASAP**.

When can you enable PQC?

Also customers:



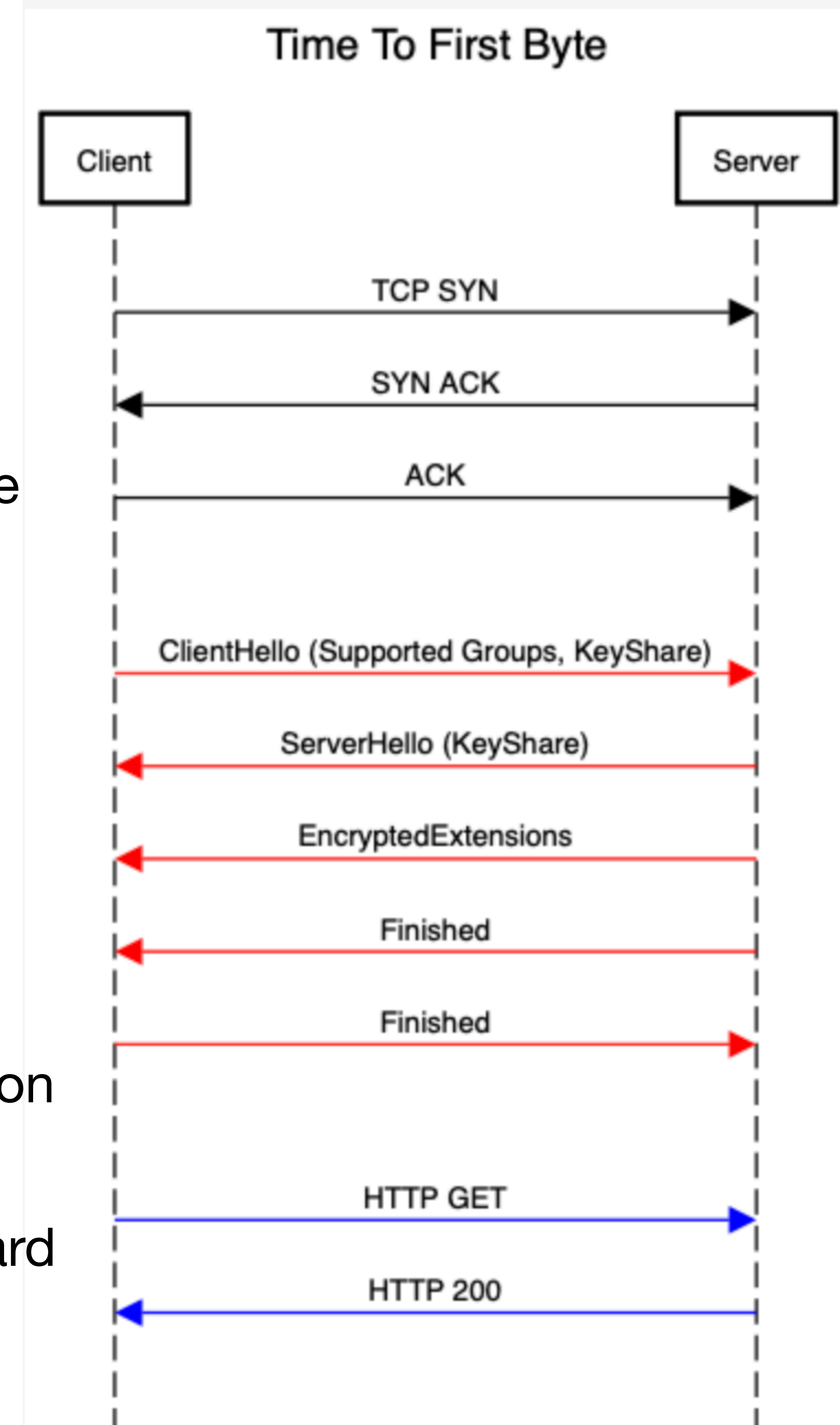
Don't just **enable** PQC!

We need several months to test.

Lessons

Performance

- Key generation, encap/decap, signing performance all negligible impact (comparable to EC, *better* than RSA)
- Yes, ClientHello will increase in size, but:
 - impact is amortized for persistent connections (e.g., Akamai-to-Origin)
 - browsers already carry half of that cost whether you enable PQC or not
 - typical HTTP traffic is (comparatively) large; impact primarily on TTFB, not user experience
 - cost of PQC is simply cost of HTTPS on the web going forward



Lessons

Compatibility

- All major browsers since early 2025 the latest (but a lot fewer of the bots)
- Most TLS libraries, toolkits, programming languages released within the last year.
- Web servers, proxies, side-cars etc. reaching broad adoption in 2026
- About 70% of (human) client traffic uses PQC
- About 44% of Top 150K domains support PQC using X25519MLKEM768
 - only Amazon and Squarespace support SecP256r1MLKEM768
 - only Google supports MLKEM1024

Recommendations

- Uhm... upgrade to TLS 1.3?
- Upgrade your libraries / stacks; chances are, you get PQC enabled by default already!
- Ensure non-PQC origins can handle a hybrid ClientHello.
- Enable PQC on the edge regardless of your origin readiness.
- Make it enabled *by default*.
- Use a lot of buzzwords when talking to executives!
- Tone down the buzzwords when talking to engineers.

Takeaways

- Adopting PQC is slow and painful, but not because PQC is uniquely difficult to deploy.
- Lack of cryptographic agility and regular software updates are the biggest blockers.
- Updating to PQC Kex is now easy / you get it for free.
- Certificates will be a whole different story.
- You really want to start working on support for SVCB DNS records.
- Don't get me started on DNSSEC.

And of course remember...



**Cryptography is typically bypassed,
not penetrated.**



Thanks!

Private

- Email: jschauma@netmeister.org
- Mastodon: [@jschauma@mstdn.social](https://mstdn.social/@jschauma)
- Web: <https://www.netmeister.org/>

Akamai:

- jschauma@akamai.com
- <https://www.akamai.com/blog?author=jan-schaumann>
- <https://techdocs.akamai.com/property-mgr/docs/pqc-origin>
- <https://techdocs.akamai.com/property-mgr/docs/pqc-client-to-edge>

Links

- "Quantum Computers Are Not a Threat to 128-bit Symmetric Keys" <https://words.filippo.io/128-bits/>
- "Quantum frontiers may be closer than they appear": <https://blog.google/innovation-and-ai/technology/safety-security/cryptography-migration-timeline/>
- "Shor's algorithm is possible with as few as 10,000 reconfigurable atomic qubits": <https://arxiv.org/pdf/2603.28627>
- "Securing Elliptic Curve Cryptocurrencies against Quantum Vulnerabilities: Resource Estimates and Mitigations": <https://arxiv.org/abs/2603.28846>
- "Safeguarding cryptocurrency by disclosing quantum vulnerabilities responsibly": <https://research.google/blog/safeguarding-cryptocurrency-by-disclosing-quantum-vulnerabilities-responsibly/>
- "Sites using PQC (February 2026)": <https://www.netmeister.org/blog/pqc-use-2026-02.html>
- "Post-Quantum Cryptography Beyond TLS: Remain Quantum Safe": <https://www.akamai.com/blog/security/post-quantum-cryptography-beyond-tls>
- "Cultivating a robust and efficient quantum-safe HTTPS": <https://blog.google/security/cultivating-a-robust-and-efficient-quantum-safe-https/>
- "Keeping the Internet fast and secure: introducing Merkle Tree Certificates" <https://blog.cloudflare.com/bootstrap-mtc/>
- "A Guide to International Post-Quantum Cryptography Standards": <https://www.akamai.com/blog/security/guide-international-post-quantum-cryptography-standards>
- "TLS 1.3 Hybrid Key Exchange using X25519Kyber768 / ML-KEM": <https://www.netmeister.org/blog/tls-hybrid-kex.html>
- "Merkle Tree Certificates": <https://datatracker.ietf.org/doc/draft-ietf-plants-merkle-tree-certs/>

PQC Adoption beyond TLS

- OpenSSH defaults to X25519MLKEM768 since 9.9, warns about non-PQ since 10.1
- TLS-based VPNs may be able to use X25519MLKEM768; separate drafts for hybrid keys in IPSec/IKE
- Signal & WhatsApp use the Sparse Post-Quantum Ratchet (SPQR); iMessages use Apple's PQ3
- DNSSEC faces size challenges; possible solution in Merkle Tree Ladder Mode signatures (SLH-DSA-MTL)