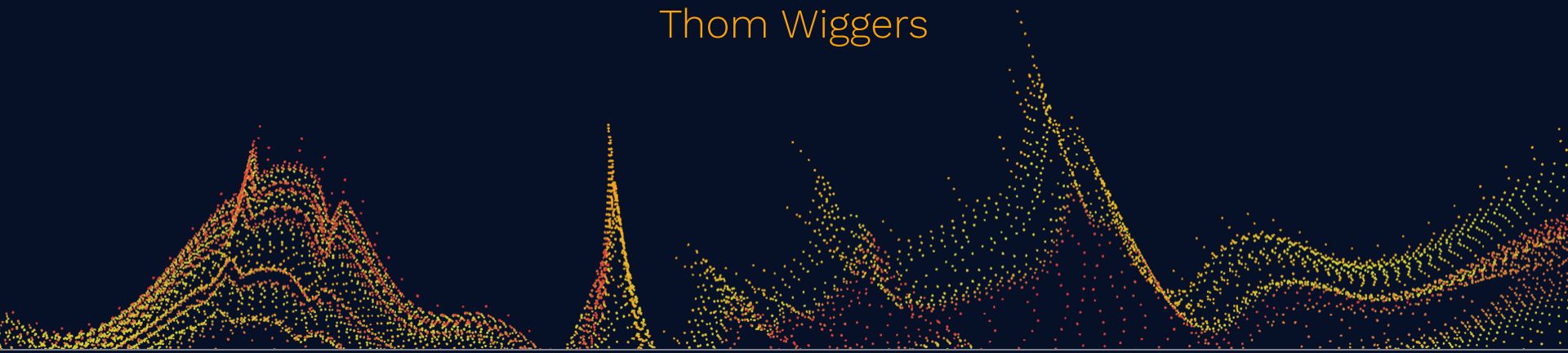
Fully PQ TLS in the WWW

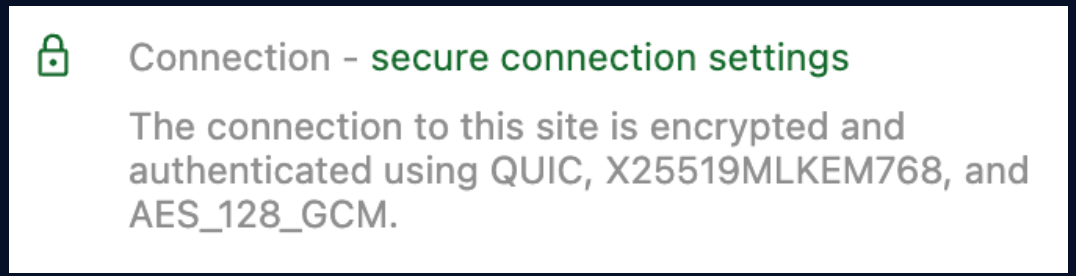
Where we're at and where we're going

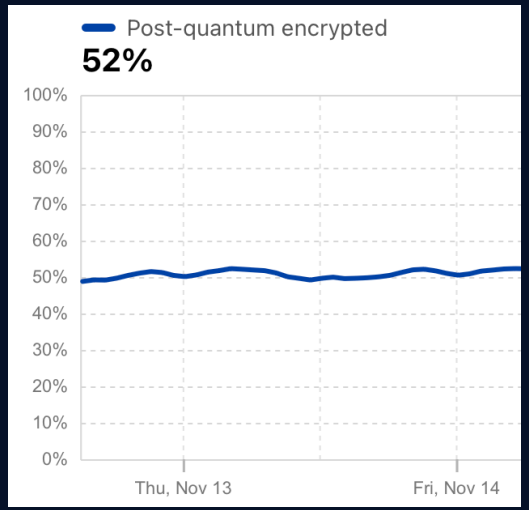
Thom Wiggers



PQ is going great

Cloudflare, Google Chrome, Microsoft Edge, and Firefox deployed PQ key exchange in web browsing.





radar.cloudflare.com

To Do: Where we're at

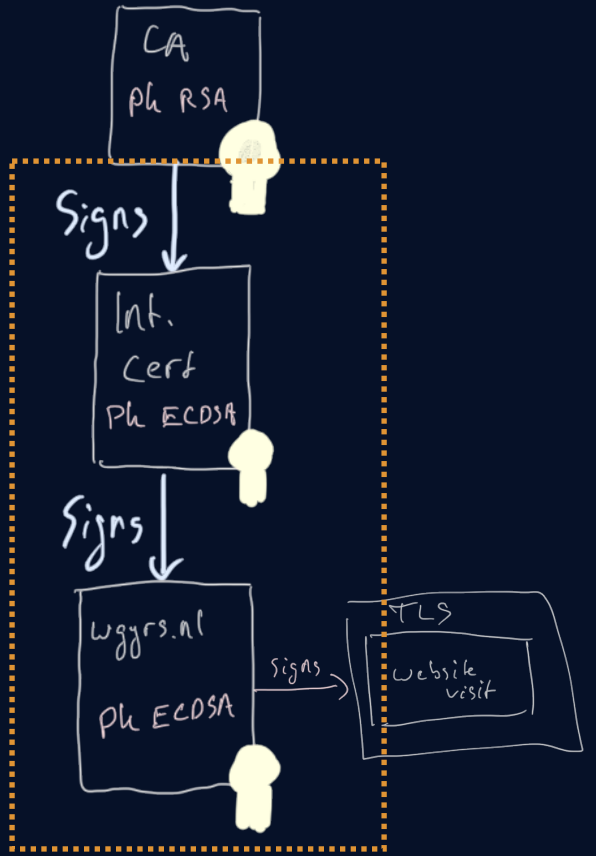
To Do: Where we're at

☑ Get post-quantum key exchange

To Do: Where we're at

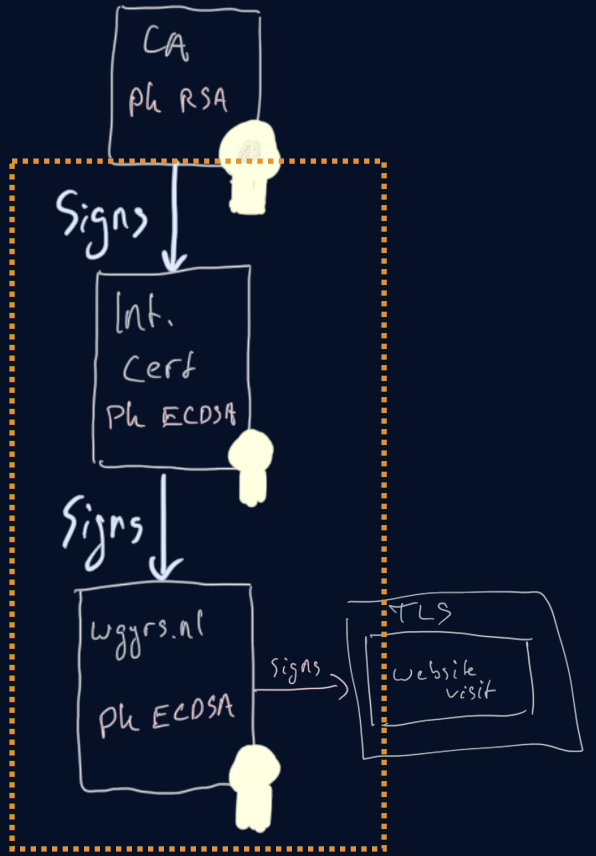
- ☑ Get post-quantum key exchange
- ☐ Get post-quantum authentication

Authentication in TLS



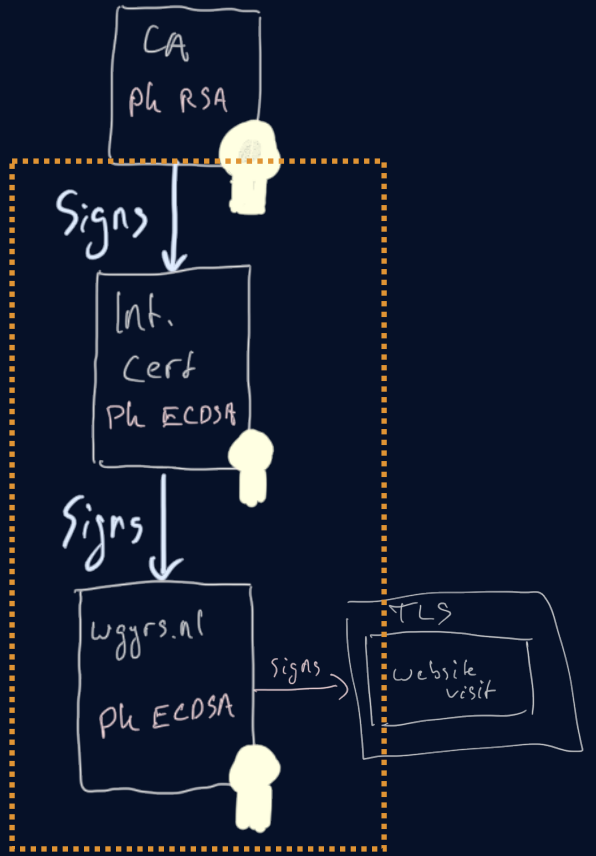
Authentication in TLS

- There are many signatures in TLS



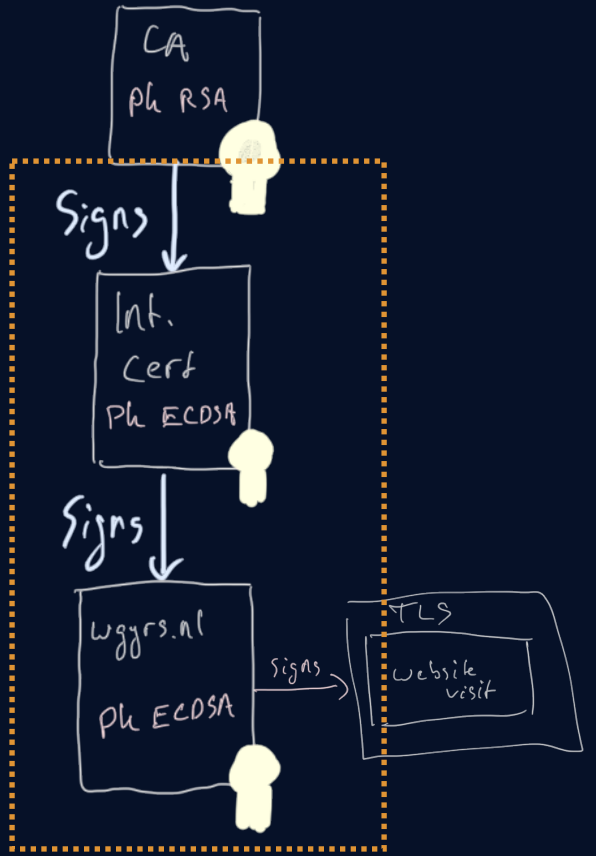
Authentication in TLS

- There are many signatures in TLS
- Server certificate:
  - 1x public key, 1x signature
  - Bonus: 3x Certificate Transparency SCTs
  - 1x Signature on handshake



Authentication in TLS

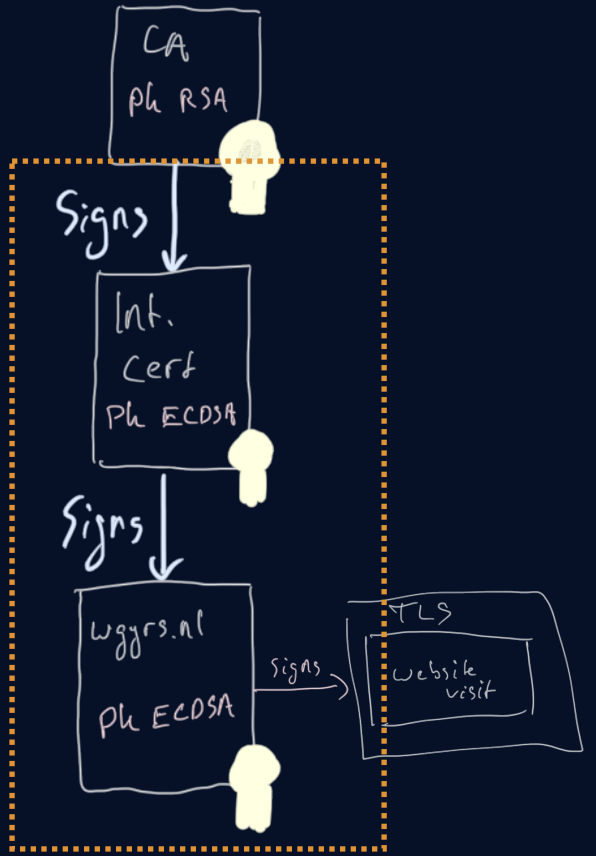
- There are many signatures in TLS
- Server certificate:
  - 1x public key, 1x signature
  - Bonus: 3x Certificate Transparency SCTs
  - 1x Signature on handshake
- Intermediate Certificate:
  - 1x public key, 1x signature



Authentication in TLS

- There are many signatures in TLS
- Server certificate:
  - 1x public key, 1x signature
  - Bonus: 3x Certificate Transparency SCTs
  - 1x Signature on handshake
- Intermediate Certificate:
  - 1x public key, 1x signature

Replacing all of this by ML-DSA adds 18–36 kB!



Point of View: website operator

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse
- Requires updating certificate infra

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse
- Requires updating certificate infra
- Dependencies on suppliers and ecosystem

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse
- Requires updating certificate infra
- Dependencies on suppliers and ecosystem
- ~4% slowdown for ~2kB means big slowdown for ~18kB

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse
- Requires updating certificate infra
- Dependencies on suppliers and ecosystem
- ~4% slowdown for ~2kB means big slowdown for ~18kB

(See also last year's talk)

Point of View: website operator

PQ key exchange:

- Solves Harvest-Now-Decrypt-Later
- Only need to update in 1 spot
- Well-tested now
- Adds ~2kB of data to handshake
- ~4% slowdown for most clients is costly but acceptable (src: Google)

PQ authentication:

- Will only protect after quantum apocalypse
- Requires updating certificate infra
- Dependencies on suppliers and ecosystem
- ~4% slowdown for ~2kB means big slowdown for ~18kB

"I'll wait"

(See also last year's talk)

Certificate Transparency

- Certificate transparency is a public log of all issued certificates
  - In particular, a Merkle Tree
- Aim: detect “DigiNotar” incidents
  - Hacked CA issued certificates for gmail.com
- Chrome, Firefox, Safari require Certificate Transparency

Certificate Transparency is fragile

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard
  - Many failure modes lead to log disqualification

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard
  - Many failure modes lead to log disqualification
- Running a log is expensive

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard
  - Many failure modes lead to log disqualification
- Running a log is expensive
  - ~20 TB disk per year

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard
  - Many failure modes lead to log disqualification
- Running a log is expensive
  - ~20 TB disk per year
  - ~10 TB per day bandwidth

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency is fragile

- Running a log is hard
  - Many failure modes lead to log disqualification
- Running a log is expensive
  - ~20 TB disk per year
  - ~10 TB per day bandwidth
  - Before PQC

Source: Let's Encrypt presentation in PLANTS meeting at IETF 124

Certificate Transparency uses Trees

- CT logs only accept submissions from trusted CAs
  - The list only contains validated certificates
- Merkle Trees can produce a proof of inclusion in the tree
- So a proof of inclusion could prove validity of the certificate!
  - Inclusion proof is just a bunch of hashes

MTC: Merkle Tree Certificates

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs
  - CT was layered on top of existing PKI

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs
  - CT was layered on top of existing PKI
  - Complications stem from the gap between issuance and logging

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs
  - CT was layered on top of existing PKI
  - Complications stem from the gap between issuance and logging
- A Merkle Tree CA first logs, then signs a checkpoint and collects cosignatures

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs
  - CT was layered on top of existing PKI
  - Complications stem from the gap between issuance and logging
- A Merkle Tree CA first logs, then signs a checkpoint and collects cosignatures
  - Each CA runs a separate issuance log

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Merkle Tree Certificates

- A traditional CA first signs, then logs the result and collects SCTs
  - CT was layered on top of existing PKI
  - Complications stem from the gap between issuance and logging
- A Merkle Tree CA first logs, then signs a checkpoint and collects cosignatures
  - Each CA runs a separate issuance log
  - Independent mirrors and witnesses co-sign log state

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Issuance and Verification

- CA creates certificates and builds Merkle Tree
- Tree state gets checked and tree head is signed by co-signers
- Certificate is now:
  - Public key
  - Path in Merkle Tree
  - The (co)signatures on the tree head

- Note that these signed tree heads are shared with many log entries, and can be distributed out-of-band!

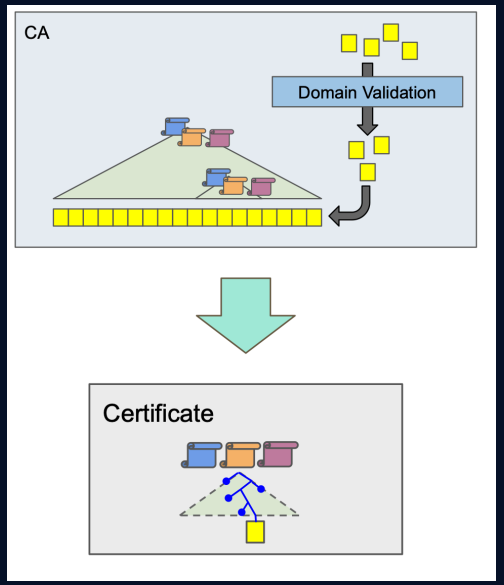


Image Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: Transparency sustainability

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44
- Mirrors now provably identical

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44
- Mirrors now provably identical
  - monitors only need to download one time

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44
- Mirrors now provably identical
  - monitors only need to download one time
- More feasible to run mirrors

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44
- Mirrors now provably identical
  - monitors only need to download one time
- More feasible to run mirrors
  - Load spread out more

MTC: Transparency sustainability

- Instead of many distinct logs with all certificates, CA now maintains its one list
- Cosigners ensure transparency properties
- Compared to CT today, MTC log entries are 12x smaller
  - 97x smaller than CT+ML-DSA-44
- Mirrors now provably identical
  - monitors only need to download one time
- More feasible to run mirrors
  - Load spread out more
- Log errors/downtime only means issuance failure, no log disqualification

MTC: Optimised certificates

- 3 ML-DSA signatures: 7,260 bytes (1 CA sig, 2 SCTs)
- 1 inclusion proof: 736 bytes
  - Tree size depends on issuance rate
  - (estimated from Web PKI / Let's Encrypt volumes)
- No longer a need for intermediate certificates

Src: https://datatracker.ietf.org/meeting/124/materials/slides-124-plants-solution-space-and-dispatched-work-00

MTC: How to deploy this?

- TLS Clients (browsers) must be updated with support
  - Ability to fetch tree heads out-of-band
- TLS Servers must be updated with support
  - Clients will indicate which tree-heads they have
  - Server selects “full” or “signature-less” MTC certificate
    - “Full” will include the signatures on the tree head
    - “Signature-less” omits signatures if client is up-to-date with tree heads
  - Server must update its inclusion proof periodically
    - Likely via ACME (aka Let’s Encrypt’s “certbot”)

What about “old-fashioned” certificates

- MTC will (likely) only work in WebPKI and similar settings
- “Old-fashioned” certificates with chains of signatures will probably continue to exist
  - Fall-back in the WebPKI
  - Private PKIs
  - Non-Web use cases

How real is this?

- Proposal is being pushed by Google Chrome and Cloudflare
- Standards are being developed in the PLANTS (PKI, Logs, And Tree Signatures) working group in IETF
  - Disclaimer: I was asked to be co-chair of this working group
  - Please review and comment
  - Join the mailing list plants@ietf.org

- Cloudflare and Google announced they will start experimenting on the web in 2026

More lessons from real people, on our podcast

- **Shielded - the Last Line of Cyber Defense Podcast** launched March 2025 - now with nearly 160,000 subscribers on YouTube!
- Purpose - to share PQC migration stories from global organisations
- 21 episodes so far, including:
  - Cloudflare, CISA, Linux Foundation, NIST, Signal Messenger, Thales, Entrust, Lattice Semiconductor, UK NCSC, Schneider Electric, Bill Buchanan, AMD, Yolanda Reid, Digicert, OpenSLL Foundation, Capgemini, HSBC, HP and many more

YouTube

Spotify

Apple