PQConnect
Automated Post-Quantum End-to-End Tunnels

Daniel J. Bernstein, Tanja Lange, Jonathan Levin, Bo-Yin Yang

27 December 2024

Urgency of moving to post-quantum cryptography

White House briefing urges move to PQC.
Deadline: 2035.

2024 EU PQC transition roadmap (link)

COMMISSION RECOMMENDATION

of 11.4.2024

on a Coordinated Implementation Roadmap for the transition to Post-Quantum Cryptography

(5) Member States should consider migrating their current digital infrastructures and services for public administrations and other critical infrastructures to Post-Quantum Cryptography as soon as possible, inducing a fundamental shift in cryptographic algorithms, protocols and systems. As highlighted in the Commission's recent White Paper "How to master Europe's digital infrastructure needs", this requires a coordinated effort involving government agencies, standardization bodies, industry stakeholders, researchers and cybersecurity professionals.

(9) Member States and the Union should continue to cooperate actively with their international strategic partners in the development of international standards in Post-Quantum Cryptography with a view to ensuring interoperability of communications going forward.

27 Nov 2024: Joint statement by 17 EU countries (Link)

Securing Tomorrow, Today: Transitioning to Post-Quantum Cryptography

> To this end, a Work Stream on PQC, co-chaired by France, Germany and the Netherlands, has been created as part of the NIS Cooperation Group following a recommendation [9] of the European Commission. **We encourage active engagement from all EU member states in this work stream** throughout the process of preparing a roadmap for the transition to Post-Quantum Cryptography to ensure the quantum resilience of the European Union's digital infrastructures.

[..]

27 Nov 2024: Joint statement by 17 EU countries (Link)

Securing Tomorrow, Today: Transitioning to Post-Quantum Cryptography

> To this end, a Work Stream on PQC, co-chaired by France, Germany and the Netherlands, has been created as part of the NIS Cooperation Group following a recommendation [9] of the European Commission. **We encourage active engagement from all EU member states in this work stream** throughout the process of preparing a roadmap for the transition to Post-Quantum Cryptography to ensure the quantum resilience of the European Union's digital infrastructures.

[..]

we recommend that these should be protected against 'store now, decrypt later' attacks as soon as possible, latest by the end of 2030. Moreover, we also recommend to develop detailed transition plans for public-key infrastructure systems in the same timeframe.

Australian Government

Australian Signals Directorate

ASD AUSTRALIAN SIGNALS DIRECTORATE

ACSC Australian Cyber Security Centre

Information Security Manual

Last updated: December 2024

Guidelines for Cryptography

Disallows pre-quantum by 2030

Store now, decrypt later

You got me convinced,

I want to move.

Tell me what to do!

Wait!

Wait?

The NIST standards just came out

No PQC for your application

But my data . . . ?

But everybody's data . . . ?

PQConnect

- Do not patch PQC onto existing network protocols, but add a new layer with superior security.
- Can be gradually deployed – do your part today!
- Install PQConnect to add support for VPN-like tunnels to your laptop. If you are a system administrator, install it on your servers.
- Every connection will be automatically protected if the laptop and server support PQConnect thanks to automatic peer discovery.
- PQConnect is designed for security. Handshake proven using Tamarin prover (formal verification tool).
- Use Curve25519 (pre-quantum) and Classic McEliece (conservative PQC) for long-term identity keys.
- Use Curve25519 (pre-quantum) and Streamlined NTRU Prime (PQC) for ephemeral keys.

PQConnect Features

What's in the box?
Network software that:

1. Provides *application-independent* Post-Quantum network data protection
   - Most applications protected with no extra configuration.
   - Integrates seamlessly with security protocols higher in the stack (e.g., TLS)
2. Backward-compatible ("Move fast, but let's not break things")
   - Automatic peer discovery from advertisements sent to users through DNS.
   - Non-PQConnect users experience no disruptions.
3. Adds extra cryptographic protections where none currently exist
   - Entire packet encryption, providing header confidentiality
   - Packets without other cryptographic protection (e.g., no TLS) now have it.
4. Client and Server software currently available for GNU/Linux

Data Flow: Server Identification

How do clients discover servers?

Typical DNS query: `bench.cr.yp.to IN A`?

- `bench.cr.yp.to`: query name
- `A`: query type (IPv4 address)

Data Flow: Server Identification

How do clients discover servers?

Typical DNS query: `bench.cr.yp.to IN A`?

- `bench.cr.yp.to`: query name
- `A`: query type (IPv4 address)

Typical DNS response: `bench.cr.yp.to IN A 131.193.32.110`

- `bench.cr.yp.to`: response name
- `131.193.32.110` : response data

Data Flow: Server Identification

Sometimes a bit more complicated.

- DNS query:
  `www.amazon.com IN A?`

Data Flow: Server Identification

Sometimes a bit more complicated.

- DNS query:
  `www.amazon.com IN A?`
- Response:
  ```
  www.amazon.com.  IN CNAME tp.[...].amazon.com.
  tp.[...].amazon.com.  IN CNAME www.amazon.com.edgekey.net.
  www.amazon.com.edgekey.net.  IN CNAME [...].akamaiedge.net.
  [...].akamaiedge.net.  IN A 23.199.25.236
  ```

Data Flow: Server Identification

Sometimes a bit more complicated.

- DNS query:
  `www.amazon.com IN A?`
- Response:
  ```
  www.amazon.com.  IN CNAME tp.[...].amazon.com.
  tp.[...].amazon.com.  IN CNAME www.amazon.com.edgekey.net.
  www.amazon.com.edgekey.net.  IN CNAME [...].akamaiedge.net.
  [...].akamaiedge.net.  IN A 23.199.25.236
  ```

Client follows the chain to find IP address.

Data Flow: Server Identification

Sometimes a bit more complicated.

- DNS query:
  `www.amazon.com IN A?`
- Response:
  ```
  www.amazon.com.  IN CNAME tp.[...].amazon.com.
  tp.[...].amazon.com.  IN CNAME www.amazon.com.edgekey.net.
  www.amazon.com.edgekey.net.  IN CNAME [...].akamaiedge.net.
  [...].akamaiedge.net.  IN A 23.199.25.236
  ```

Client follows the chain to find IP address.
Ignores intermediate records.

Data Flow: Server Identification

But we don't *have* to ignore them!

- DNS query:
  `iis.sinica.edu.tw IN A?`

Data Flow: Server Identification

But we don't *have* to ignore them!

- DNS query:
  `iis.sinica.edu.tw IN A?`
- Response:
  `iis.sinica.edu.tw.  IN CNAME IUsePQConnect.sinica.edu.tw.`
  `IUsePQConnect.sinica.edu.tw.  IN A 140.109.20.229`

Data Flow: Server Identification

But we don't *have* to ignore them!

- DNS query:
  `iis.sinica.edu.tw IN A?`
- Response:
  `iis.sinica.edu.tw. IN CNAME IUsePQConnect.sinica.edu.tw.`
  `IUsePQConnect.sinica.edu.tw. IN A 140.109.20.229`

Or better:

- Response:
  `iis.sinica.edu.tw. IN CNAME <pqc-public-key>.sinica.edu.tw.`
  `<pqc-public-key>.sinica.edu.tw. IN A 140.109.20.229`

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Kernel view:

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Kernel view:

- `IP 140.109.20.229.443 > 192.168.81.142.56068`? Accept.

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Kernel view:

- `IP 140.109.20.229.443 > 192.168.81.142.56068`? Accept.
- `IP 131.155.68.89.624 > 192.168.81.142.42423`? Accept.

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Kernel view:

- `IP 140.109.20.229.443 > 192.168.81.142.56068`? Accept.
- `IP 131.155.68.89.624 > 192.168.81.142.42423`? Accept.
- `IP 168.95.1.1.53 > 192.168.81.142.59959`?
  Port 53! Send to PQConnect!

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Kernel view:

- `IP 140.109.20.229.443 > 192.168.81.142.56068`? Accept.
- `IP 131.155.68.89.624 > 192.168.81.142.42423`? Accept.
- `IP 168.95.1.1.53 > 192.168.81.142.59959`?
  Port 53! Send to PQConnect!

Currently PQConnect filters on port 53.
There are other ways to intercept DNS responses (e.g., PQConnect could run its own DNS resolver, hook into systemd-resolved, etc.)

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

PQConnect view:

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

PQConnect view:

- `IP 168.95.1.1.53 > 192.168.81.142.54712`
  `bench.cr.yp.to IN A 131.193.32.110`

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

PQConnect view:

- `IP 168.95.1.1.53 > 192.168.81.142.54712`
  `bench.cr.yp.to IN A 131.193.32.110`

  OK. Not interesting. Send it back

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

PQConnect view:

- `IP 168.95.1.1.53 > 192.168.81.142.54712`
  `bench.cr.yp.to IN A 131.193.32.110`

  OK. Not interesting. Send it back
- `IP 168.95.1.1.53 > 192.168.81.142.59959`
  `www.pqconnect.net.  IN CNAME pq1[...].pqconnect.net.`
  `pq1[...].pqconnect.net.  60 IN A 131.155.69.126`

  NICE! We found a supporting server. Rewrite 131.155.69.126 to a local address that routes to PQConnect (e.g., 10.59.0.2)

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Application view:

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Application view:

- `('131.193.32.110', 80) = getaddrinfo('bench.cr.yp.to', 80)`
  Great. Send TCP handshake to 131.193.32.110

Capturing Application Traffic

PQConnect inserts itself into the network stack to inspect incoming DNS responses.

Application view:

- `('131.193.32.110', 80) = getaddrinfo('bench.cr.yp.to', 80)`
  Great. Send TCP handshake to 131.193.32.110
- `('10.59.0.2', 80) = getaddrinfo('www.pqconnect.net', 80)`
  Great. Send TCP handshake to 10.59.0.2

Connection now routed through PQConnect

Demo

Protocol Overview - Key Distribution

PQConnect establishes a secure channel using:

- Long term keys: Classic McEliece (post-quantum), X25519 (pre-quantum)
- Ephemeral keys: Streamlined NTRU Prime (post-quantum), X25519

Protocol Overview - Key Distribution

PQConnect establishes a secure channel using:

- Long term keys: Classic McEliece (post-quantum), X25519 (pre-quantum)
- Ephemeral keys: Streamlined NTRU Prime (post-quantum), X25519

Classic McEliece (mceliece6960119) public keys are 1047319 bytes.
Doesn't fit in a DNS record :(

Protocol Overview - Key Distribution

PQConnect establishes a secure channel using:

- Long term keys: Classic McEliece (post-quantum), X25519 (pre-quantum)
- Ephemeral keys: Streamlined NTRU Prime (post-quantum), X25519

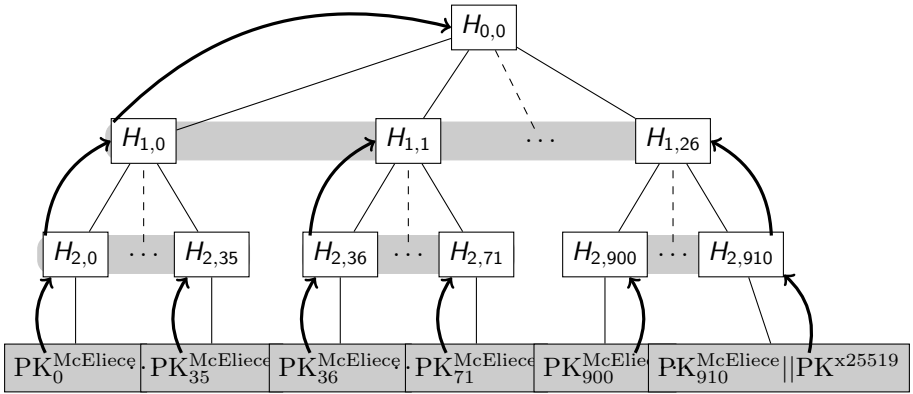
Classic McEliece (mceliece6960119) public keys are 1047319 bytes.
Doesn't fit in a DNS record :(

Advertise the hash of the long term keys instead

Protocol Overview - Key Distribution

Distribute long term keys from a key server as a Merkle Tree.

Clients request packets from each layer of the tree, verifying their hash against the parent node.



Protocol Overview - Handshake

PQConnect establishes a secure channel using:

- Long term keys: Classic McEliece (post-quantum), X25519 (pre-quantum)
- Ephemeral keys: Streamlined NTRU Prime (post-quantum), X25519

Protocol Overview - Handshake

PQConnect establishes a secure channel using:

- Long term keys: Classic McEliece (post-quantum), X25519 (pre-quantum)
- Ephemeral keys: Streamlined NTRU Prime (post-quantum), X25519

Client computes secret against all four public keys.
Sends handshake M to server.

Server receives M, decapsulates, etc., and computes same shared secret.

Handshake security

Hybrid approach

4 key agreements take place:

- 2 long-term (post- and pre-quantum)
- 2 ephemeral (post- and pre-quantum)

Each PKC scheme layered "inside of" the next.
Forces sequential attack to obtain innermost keys/ciphertexts.

PQ cryptography protects against quantum attackers.
X25519 assures that security is no worse than current state-of-the-art

Handshake security

Hybrid approach

4 key agreements take place:

- 2 long-term (post- and pre-quantum)
- 2 ephemeral (post- and pre-quantum)

Each PKC scheme layered "inside of" the next.
Forces sequential attack to obtain innermost keys/ciphertexts.

PQ cryptography protects against quantum attackers.
X25519 assures that security is no worse than current state-of-the-art

Formal security proof

Security properties of the handshake were formally proven using Tamarin Prover[1].

Handshake security

Hybrid approach

4 key agreements take place:

- 2 long-term (post- and pre-quantum)
- 2 ephemeral (post- and pre-quantum)

Each PKC scheme layered "inside of" the next.
Forces sequential attack to obtain innermost keys/ciphertexts.

PQ cryptography protects against quantum attackers.
X25519 assures that security is no worse than current state-of-the-art

Formal security proof

Security properties of the handshake were formally proven using Tamarin Prover[1].

[1] https://tamarin-prover.github.io/

Protocol Overview - Symmetric Crypto

Each packet sent between peers encrypted with unique key.
How long to keep keys in memory?

- When sending packet, erase key immediately after sending
- When receiving packet... ?

Protocol Overview - Symmetric Crypto

Each packet sent between peers encrypted with unique key.
How long to keep keys in memory?

- When sending packet, erase key immediately after sending
- When receiving packet... ?

Packets can be delayed, dropped, reordered.

Protocol Overview - Symmetric Crypto

Each packet sent between peers encrypted with unique key.
How long to keep keys in memory?

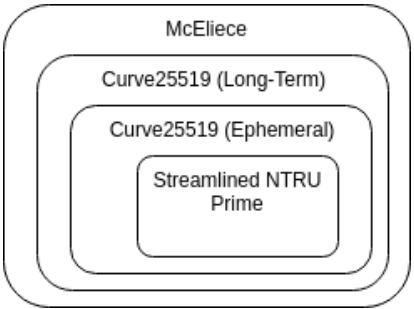
- When sending packet, erase key immediately after sending
- When receiving packet... ?

Packets can be delayed, dropped, reordered.

Our approach: Delete keys when packets arrive, or at the latest after 2 min.

PQConnect handshake: Nesting schemes

Most conservative system on the outside.



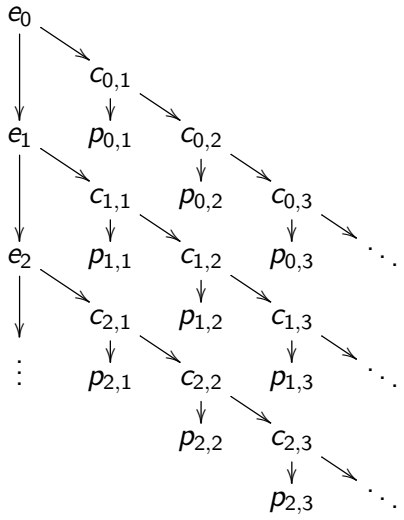
Attacker can see long-term Curve25519 identity key, can break it with a quantum computer, but cannot obtain DH value as client's share is wrapped.

Key ratchet advances by message and time

e_0 is the initial key.
Immediately advance ratchet in 3 ways:

- New root epoch key: e_1.
- New chain keys: $c_{0,1}, c_{0,2}$.
- New packet key: $p_{0,1}$.

New epoch every 30 seconds.
Keys erased upon use, or at latest after 2 minutes.
Packet keys can deal with delayed transmissions.



Tell me what to do!

Run PQConnect on your devices!

Run PQConnect on your server!

More information

- Visit `https://www.pqconnect.net/` for more information and software download.
- Join `https://zulip.pqconnect.net/` to discuss the project and get updates.