PQConnect: Automated post-quantum end-to-end tunnels

Daniel J. Bernstein

Joint work with:

Tanja Lange, Jonathan Levin, Bo-Yin Yang

CVE-2025-8393

Regarding Dreame's robot vacuums, which had received ETSI security certification: "A TLS vulnerability exists in the phone application used to manage a connected device. The phone application accepts self-signed certificates when establishing TLS communication which may result in man-in-the-middle attacks on untrusted networks. Captured communications may include user credentials and sensitive session tokens."

CVE-2025-6037

"Vault and Vault Enterprise ('Vault') TLS certificate auth method did not correctly validate client certificates when configured with a non-CA certificate as trusted certificate. In this configuration, an attacker may be able to craft a malicious certificate that could be used to impersonate another user."

CVE-2025-58126

"Improper Certificate Validation in Checkmk Exchange plugin VMware vSAN allows attackers in MitM position to intercept traffic."

CVE-2025-52497

"Mbed TLS before 3.6.4 has a PEM parsing one-byte heap-based buffer underflow, in `mbedtls_pem_read_buffer` and two `mbedtls_pk_parse` functions, via untrusted PEM input."

CVE-2025-52496

"Mbed TLS before 3.6.4 has a race condition in AESNI detection if certain compiler optimizations occur. An attacker may be able to extract an AES key from a multithreaded program, or perform a GCM forgery."

CVE-2025-49812

"In some `mod_ssl` configurations on Apache HTTP Server versions through to 2.4.63, an HTTP desynchronisation attack allows a man-in-the-middle attacker to hijack an HTTP session via a TLS upgrade. Only configurations using 'SSLEngine optional' to enable TLS upgrades are affected."

CVE-2025-41255

"Cyberduck and Mountain Duck improperly handle TLS certificate pinning for untrusted certificates (e.g., self-signed), unnecessarily installing it to the Windows Certificate Store of the current user without any restrictions."

CVE-2025-36005

"IBM MQ Operator LTS 2.0.0 through 2.0.29, MQ Operator CD 3.0.0, 3.0.1, 3.1.0 through 3.1.3, 3.3.0, 3.4.0, 3.4.1, 3.5.0, 3.5.1, 3.6.0, and MQ Operator SC2 3.2.0 through 3.2.13 Internet Pass-Thru could allow a malicious user to obtain sensitive information from another TLS session connection by the proxy to the same hostname and port due to improper certificate validation."

CVE-2025-32407

"Samsung Internet for Galaxy Watch version 5.0.9, available up until Samsung Galaxy Watch 3, does not properly validate TLS certificates, allowing for an attacker to impersonate any and all websites visited by the user. This is a critical misconfiguration in the way the browser validates the identity of the server. It negates the use of HTTPS as a secure channel, allowing for Man-in-the-Middle attacks, stealing sensitive information or modifying incoming and outgoing traffic."

CVE-2025-31972

"HCL BigFix SM is affected by a Sensitive Information Exposure vulnerability where internal connections do not use TLS encryption which could allow an attacker unauthorized access to sensitive data transmitted between internal components."

TLS is a very large programming project

Broad TLS deployment requires integrating TLS into many protocols and many more applications. Some ways to see how big this project is:

- Wide range of applications in TLS CVEs.

TLS is a very large programming project

Broad TLS deployment requires integrating TLS into many protocols and many more applications. Some ways to see how big this project is:

- Wide range of applications in TLS CVEs.
- Different languages for applications drive demand for TLS libraries in many languages.

TLS is a very large programming project

Broad TLS deployment requires integrating TLS into many protocols and many more applications. Some ways to see how big this project is:

- Wide range of applications in TLS CVEs.
- Different languages for applications drive demand for TLS libraries in many languages.
- GitHub search for "SSL" finds >2 million PRs. Spot-checks indicate that most of these really are SSL, not something else by that name.

TLS is a very large programming project

Broad TLS deployment requires integrating TLS into many protocols and many more applications. Some ways to see how big this project is:

- Wide range of applications in TLS CVEs.
- Different languages for applications drive demand for TLS libraries in many languages.
- GitHub search for "SSL" finds >2 million PRs. Spot-checks indicate that most of these really are SSL, not something else by that name.
- HTTPS percentage in Firefox web-page loads: 30% in 2015, 80% in 2020, 80% in 2025. The other 20%: No TLS code? Not configured?

We have a security problem right now

Large-scale attackers are recording ciphertexts,
planning to break them with quantum computers.

We have a security problem right now

Large-scale attackers are recording ciphertexts, planning to break them with quantum computers.

TLS response:

- Upgrade to post-quantum TLS.
- Keep trying to fix TLS vulnerabilities.
- For all of the data that isn't even encrypted today, keep trying to increase TLS deployment.

We have a security problem right now

Large-scale attackers are recording ciphertexts,
planning to break them with quantum computers.

TLS response:

- Upgrade to post-quantum TLS.
- Keep trying to fix TLS vulnerabilities.
- For all of the data that isn't even encrypted today, keep trying to increase TLS deployment.

While this gigantic TLS project is continuing,
is there anything else we can do to protect users?

An easier path: VPNs

VPNs have a big software-engineering advantage: they protect unmodified applications!

A VPN client typically routes all outgoing traffic through an encrypted tunnel to a proxy.
Proxy is specified in the VPN config.

An easier path: VPNs

VPNs have a big software-engineering advantage: they protect unmodified applications!

A VPN client typically routes all outgoing traffic through an encrypted tunnel to a proxy.
Proxy is specified in the VPN config.

VPNs supporting post-quantum crypto include Mullvad, Rosenpass, and VPNs based on OpenSSH (which in early 2022 upgraded `sntrup761` from experimental to the default KEX).

But VPNs provide incomplete protection

The client's traffic is exposed to the VPN proxy, and is exposed on network between proxy and server.

But VPNs provide incomplete protection

The client's traffic is exposed to the VPN proxy, and is exposed on network between proxy and server.

More effort:

- Add server A to your VPN config to create an end-to-end tunnel to server A.
- Add server B to your VPN config to create an end-to-end tunnel to server B.
- Etc.

But VPNs provide incomplete protection

The client's traffic is exposed to the VPN proxy, and is exposed on network between proxy and server.

More effort:

- Add server A to your VPN config to create an end-to-end tunnel to server A.
- Add server B to your VPN config to create an end-to-end tunnel to server B.
- Etc.

How often do users actually go to this effort? How do they build and maintain the lists of supporting servers?

A Boring Private Network: PQConnect

Like a VPN, PQConnect protects all applications.
Unlike a VPN, PQConnect *automatically*
builds end-to-end post-quantum tunnels
to any server that supports PQConnect.

A Boring Private Network: PQConnect

Like a VPN, PQConnect protects all applications.
Unlike a VPN, PQConnect *automatically*
builds end-to-end post-quantum tunnels
to any server that supports PQConnect.

To set up PQConnect client: Install the PQConnect software. No need for server-specific config.

Example: In Debian 13, `apt install pqconnect` installs and starts the client.

PQConnect server configuration

To set up PQConnect server:

- Install the PQConnect software.
- Publish an announcement that the server name supports PQConnect.

No need for client-specific config.

PQConnect server configuration

To set up PQConnect server:

- Install the PQConnect software.
- Publish an announcement that the server name supports PQConnect.

No need for client-specific config.

A PQConnect client connecting to that name notices the announcement, automatically builds a tunnel to the server.

How the PQConnect announcement works

Client connects to, e.g., bench.cr.yp.to. DNS:

```
bench.cr.yp.to. 30 IN CNAME
  pq1...sh273lv901sld020w020l0.cr.yp.to.

pq1...sh273lv901sld020w020l0.cr.yp.to.
  30 IN A 131.193.32.110
```

Non-PQConnect client: "Contact 131.193.32.110."
PQConnect client: "Aha, pq1... is telling me the server's PQConnect public key. I'll set up a PQConnect tunnel."

Routing data from unmodified applications

PQConnect delivers modified DNS records locally:

```
bench.cr.yp.to. 30 IN CNAME
  pq1...sh273lv901sld020w020l0.cr.yp.to.

pq1...sh273lv901sld020w020l0.cr.yp.to.
  30 IN A 10.43.0.2
```

The application sends packets to 10.43.0.2, an address managed locally by the PQConnect software. The PQConnect software encrypts the packets to send through the tunnel.

Addressing DNS forgery

Attacker forging `bench.cr.yp.to A 214.29.60.3` breaks TLS security by obtaining a Let's Encrypt certificate. Also disables PQConnect.

Addressing DNS forgery

Attacker forging `bench.cr.yp.to A 214.29.60.3` breaks TLS security by obtaining a Let's Encrypt certificate. Also disables PQConnect.

Three PQConnect response strategies:

- To the extent that DNS security tools are rolled out, they automatically protect PQConnect announcements.
- PQConnect can also be used to protect DNS, because PQConnect applies to all packets.
- PQConnect allows high-security `pq1...` links.

Fixing a problem with the TLS design

Keys are not erased *inside* a TLS session.
This is a problem if people need “forward secrecy”.

Fixing a problem with the TLS design

Keys are not erased *inside* a TLS session.
This is a problem if people need "forward secrecy".
$\Rightarrow$ Applications are pressured to keep sessions short.

Fixing a problem with the TLS design

Keys are not erased *inside* a TLS session.
This is a problem if people need "forward secrecy".
$\Rightarrow$ Applications are pressured to keep sessions short.
$\Rightarrow$ Pressure on session-startup cost.

Fixing a problem with the TLS design

Keys are not erased *inside* a TLS session.
This is a problem if people need "forward secrecy".
$\Rightarrow$ Applications are pressured to keep sessions short.
$\Rightarrow$ Pressure on session-startup cost.
$\Rightarrow$ Incentive to reduce post-quantum security levels.

Fixing a problem with the TLS design

Keys are not erased *inside* a TLS session.
This is a problem if people need "forward secrecy".
⇒ Applications are pressured to keep sessions short.
⇒ Pressure on session-startup cost.
⇒ Incentive to reduce post-quantum security levels.

PQConnect fixes this! Software erases each key
in at most 2 minutes inside a PQConnect session.
Protocol details are designed to make this work.

And another problem with the TLS design

TLS does not have long-term encryption keys.

And another problem with the TLS design

TLS does not have long-term encryption keys.
$\Rightarrow$ TLS long-term keys are only for authentication.

And another problem with the TLS design

TLS does not have long-term encryption keys.
$\Rightarrow$ TLS long-term keys are only for authentication.
$\Rightarrow$ Confidentiality comes only from keys
that TLS uses for forward secrecy.

And another problem with the TLS design

TLS does not have long-term encryption keys.
⇒ TLS long-term keys are only for authentication.
⇒ Confidentiality comes only from keys
that TLS uses for forward secrecy.
⇒ Reducing security levels for forward secrecy
(see previous slide regarding incentives) means
reducing security levels for *all* confidentiality.

And another problem with the TLS design

TLS does not have long-term encryption keys.
⇒ TLS long-term keys are only for authentication.
⇒ Confidentiality comes only from keys
that TLS uses for forward secrecy.
⇒ Reducing security levels for forward secrecy
(see previous slide regarding incentives) means
reducing security levels for *all* confidentiality.

PQConnect fixes this too! PQConnect uses
long-term KEM keys to authenticate *and* encrypt.
Also uses short-term keys for forward secrecy.

PQConnect resources

Linux software release+docs:
https://www.pqconnect.net.

Chat server: https://zulip.pqconnect.net.

Paper appeared at NDSS 2025: "PQConnect: Automated Post-Quantum End-to-End Tunnels".