Post Quantum Journey: Don't Ignore but Don't Panic

Santosh Pandit

London, July 2025

Important Disclaimers:

All views in this paper are solely mine and may not necessarily be shared by my current or past employers.

My views are based on my own experiments and may not be shared by other cyber and IT experts. This is not to invite a debate.

Always be careful when installing software in a production environment. Take full backups and thoroughly test in a non-production environment first.

All future dates (especially about software releases) are tentative and may change based on the bugs and tests run by the developer and approver communities.

Research of this nature is extremely complex and time-consuming. Omissions are to be expected. The paper is guaranteed to contain many mistakes and errors that I take ownership of.

Contents

Glossary

Acronym	**Expansion**
API	Application Programming Interface
CNSA	Commercial National Security Algorithm
ECDH	Elliptic Curve Diffie-Hellman
FIPS	Federal Information Processing Standards
IoT	Internet of Things
KEX	Key Exchange
LTS	Long-Term Support
ML-DSA	Module-Lattice-Based Digital Signature Algorithm
ML-KEM	Module-Lattice-Based Key-Encapsulation Mechanism
NIST	National Institute of Standards and Technology
OS	Operating System
PQ	Post-Quantum
PQC	Post-Quantum Cryptography
RSA	Rivest–Shamir–Adleman
SLH-DSA	Stateless Hash-Based Digital Signature Algorithm
SSH	Secure Shell
SSHv2	Secure Shell version 2
TLS	Transport Layer Security
VPN	Virtual Private Network

Executive Reading

I am sure the words “threat from quantum computers” will be mocked by some who will think of their experience in the Y2K era. Some will hope the additional investment required will be someone else’s problem after they retire. Some will ask if the problem was so big, why has nobody else done anything about it. Views will vary.

Some will argue that quantum computers are not so powerful yet and very expensive. This argument is worth exploring further. It seems to assume that in the future cyber criminals will use quantum computers to decrypt secrets in real time – tapping all communication between the sender and the receiver. That may well happen one day.

But cyber experts are worried about a different risk even more. They believe that encrypted data and communication that is stolen today can be decrypted at a later date when such quantum computers are more powerful and much cheaper. This risk is described as “harvest now, decrypt later” and from that perspective it is reasonable to say the threat of quantum computers is real and already here.

While global authorities have established milestones towards 2030 and 2035, some researchers and practitioners believe it would be unwise to wait until the last day to implement post-quantum cryptography (PQC), popularly described as quantum safe algorithms. One of the cyber experts is about to publish a book that recommends transition to PQC by 2028.

But no such transition cannot be achieved on the last day of the deadline. It requires a strategic journey that combines experimentation, performance analysis, final selection and implementation. In most cases the journey will initially rely on hybrid cryptography and in due course rely solely on quantum safe cryptography.

Good news: In my view, as of writing this note in July 2025, the building blocks of quantum safe infrastructure (namely OpenSSL and OpenSSH) are fit for purpose and contain adequate features that we can experiment with and implement hybrid cryptography. So do not ask “when should we do something about it”, instead ask “what have we done already other than holding an awareness session”. I can say this with confidence because I practice before I write.

Partially good news: Not all of the operating systems that we all use are fully ready. This means we need to strategically choose which OS will support our journey for the future. Alternatively, we need to experiment with other operating systems until such time as our chosen OS is made ready by its vendor. At least two operating systems (Debian Trixie and Arch Linux) are ready for experimentation already.

Bad news: The operating systems typically used by corporates are behind the curve. I believe their timeframe would extend to 2026 to 2027 and needs to be coupled with your own IT modernisation programs thereafter. That would be too close to the wire.

Overall, my message is, do not be complacent and do not panic either. I believe it is possible to protect ourselves by 2028, ahead of the 2030 to 2035 deadlines.

Always happy to discuss. DMs are open too.

Reading for Researchers and Practitioners

Foundation Blocks

The world's **Post Quantum Cryptography (PQ / PQC)** journey would be incomplete without its two building blocks OpenSSL and OpenSSH, so let us first take stock of those two. We will also discuss the Linux Kernel.

OpenSSL

OpenSSL is the foundation of cryptography in many operating systems. It provides cryptographic functionality required by other applications that in turn help secure our stored data or its transmission over the internet, which is inherently insecure. Without OpenSSL, we would not have been able to securely access our bank accounts over the internet.

The good news is that version 3.5 of OpenSSL, released on April 8, 2025, supports post-quantum cryptography, most notably the **National Institute of Standards and Technology (NIST)** published **Federal Information Processing Standards (FIPS)** 203, 204 and 205 algorithms. Those algorithms are likely to be officially adopted by most authorities in the world and thus form the basis of future security against the threat from quantum computers.

Annex 1 provides the details of OpenSSL's compatibility with FIPS.

Alternatives for OpenSSL

Several alternatives to OpenSSL exist, as noted on various sources like Wikipedia. However, not all are equally equipped to address the threat of quantum computers. In my view, **wolfSSL**, **LibreSSL**, and **BoringSSL** stand out as key alternatives due to their active development and specific strengths.

wolfSSL

This is a lightweight, drop-in replacement for OpenSSL, optimised for embedded systems and **Internet of Things (IoT)**. Research indicates it supports PQC, including FIPS 203 (ML-KEM), FIPS 204 (ML-DSA), and FIPS 205 (SLH-DSA), with hybrid key exchange mechanisms integrated into its **Transport Layer Security (TLS)** 1.3 stack, making it a strong candidate for quantum-resistant applications.

LibreSSL

This fork of OpenSSL by the OpenBSD project, focuses on security and simplicity by removing legacy code. However, its PQC support lags, with no full implementation of FIPS 203, 204, or 205 as of July 2025, limiting its suitability for quantum-safe use cases.

BoringSSL

Developed by Google for projects like Chrome, prioritises performance and a clean codebase. It offers experimental PQC support, including ML-KEM, but does not yet fully align with FIPS 203, 204, or 205, and its **Application Programming Interface (API)** instability may complicate adoption.

Other alternatives

Wikipedia will show you many other alternatives, such as **GnuTLS**, **mbed TLS**, **NSS**, **AWS-LC**, **Botan**, **SecureBlackbox**, **Schannel**, and **Secure Transport**, exist but vary in PQC readiness. For instance, GnuTLS and mbed TLS have experimental PQC support, while others like NSS and AWS-LC are still developing in this area. You should do your own research and contact the developers / maintainers about their roadmaps.

OpenSSH

OpenSSH leverages the functionality of OpenSSL to establish secure communication over insecure network connections. It is also the default application used in a large number of Linux based operating systems.

Here, we have partially good news. Version 10.0 of OpenSSH, released on April 9, 2025, is capable of hybrid cryptography. It enables hybrid post-quantum key exchange combining ML-KEM with classical methods like Curve25519, thus meeting FIPS 203 basics.

OpenSSH does not have in-built support for FIPS 204 and FIPS 205, but that would not stop us from establishing secure communication.

Annex 2 provides the details of OpenSSH's compatibility with FIPS.

Alternatives for OpenSSH

Several alternatives to OpenSSH exist as noted on Wikipedia and elsewhere. However, not all are equally equipped to address the threat of quantum computers. In my view, **wolfSSH** and **Dropbear** stand out as key alternatives due to their lightweight design and support for PQC.

wolfSSH

This is a lightweight SSHv2 client and server library, targeted for embedded and resource-constrained environments, serving as a viable replacement for OpenSSH in such scenarios. It supports PQC, including hybrid key exchange with algorithms like SABER, aligning with standards such as FIPS 203 for quantum-resistant key encapsulation.

Dropbear

This is a compact SSH server and client, commonly used in embedded systems and Linux distributions for its small footprint. Recent updates have added post-quantum key exchange, enabling hybrid quantum-safe connections compatible with FIPS 203 basics.

It includes post-quantum crypto by default, supporting hybrid key exchanges for quantum resistance.

Other alternatives

Then there are other names such as **libssh** (with quantum-safe forks), **lsh**, **Teleport**, **Mosh**, and commercial options exist but vary in PQC readiness. For instance, libssh has experimental PQC support via forks, while lsh and Teleport lack full implementation as of July 2025. Again, your best bet is to ask your vendor or developer about their roadmap.

Linux Kernel

The Linux Kernel can be described as the heart of a large number of operating systems and devices. The world's top 10 supercomputers use the Linux Kernel (or variants). A natural question to expect is whether the Linux Kernel is PQ ready.

There have been initial talks of improving the kernel by including PQ features. However, as of the most recent stable version 6.15.8 released on July 24, 2025, and the release-candidate 7 of the forthcoming 6.16 version, there are no meaningful features that enhance compatibility with FIPS 203, 204, and 205. Those responsibilities are currently delegated to userland, meaning individual applications should take care of their own cryptographic needs.

Over the last five years, the Linux Kernel has embedded a wide range of enhancements, including the Rust programming language and the WireGuard VPN.

Annex 3 provides an overview of the latest rc7 of the Linux Kernel.

In my personal view, it will be nice to have some post-quantum enhancements to the Linux Kernel but those are not strictly necessary for our journey, at least at this stage.

Other Kernels

Of course, we need to recognise that Linux is not the only kernel that supports all operating systems. The other noteworthy kernels include Windows NT Kernel (used in windows operating systems for servers and desktops), XNU Kernel (used in macOS for iMacs), FreeBSD, NetBSD, OpenBSD, and Solaris Kernel.

As of today, progress towards post-quantum cryptography varies: Windows and macOS have integrated PQC capabilities into their cryptographic frameworks, while the BSD variants and Solaris show limited kernel-level advancements.

Annex 4 provides an overview of PQC features in Windows and XNU kernels.

Major Operating Systems

In this section, I will focus only on major operating systems that natively integrate the foundation blocks described earlier i.e. OpenSSL and OpenSSH. We will also take stock of their PQC roadmap.

Debian

Debian's conservative approach to PQC involves community-driven integration, with experimental support in testing branches and plans for hybrid PQC by 2025.

Although the **Long-Term Support (LTS)** version 12 of Debian named Bookworm and released in 2023 uses the older versions of OpenSSL 3.0 and OpenSSH 9.2, we have good news. The next version 13 of Debian named Trixie is currently in the testing phase. It includes OpenSSL 3.5 and OpenSSH 10, enabling native FIPS 203/204/205 in OpenSSL and FIPS 203 in OpenSSH. Who knows backports may also be possible for Debian 12 in due course. I have already started using Debian 13 in production environments and started using hybrid cryptography to connect with my servers. This approach allows the use of cryptographic key pairs generated using traditional algorithms such as ed25519 but facilitates the key exchange using quantum-safe encapsulation.

Based on this successful experience, **my first prize goes to Debian Testing/Trixie.** However, please note that by its definition Debian Testing is not supposed to be used in production environment. So, if you wish to await an official release, please wait until August 9, 2025.

Arch Linux

I always find it difficult to choose between Arch Linux and Debian Testing as both of them usually have the latest versions of packages. If craving for the very latest, Arch may just win by a tiny margin. As of writing this note, Arch already has both foundation blocks (OpenSSL and OpenSSH), and they work pretty well.

I have been using Arch in production environment, but I do not expect others to take that risk. I use it only for low risk applications (i.e. static content / brochureware).

[Note: Over years, my usage of Arch Linux has reduced but that has nothing to do with the PQC. I find that most cloud service providers (and their control panel providers) are not awaropbeatre of the poor-quality templates of Arch Linux they offer to cloud customers such as myself.]

Having said everything, if you need a solid OS for PQC experimentation, **I will give second prize to Arch Linux**.

Ubuntu (Canonical)

Generally speaking, Ubuntu has some fantastic features over Debian, including the PRO and Extended Support that are particularly important to corporate needs. Also, Ubuntu's PQC roadmap is supposed to align with Debian (its upstream) and emphasises hybrid approaches for migration, with community discussions on integrating NIST standards by 2025 to 2030. However, do the actions speak as loud as words? I am disappointed.

dAs of today, we all need to await the arrival of a non-LTS release called Ubuntu Questing in October 2025 so that we can integrate our applications with OpenSSL 3.5. Even after we do that, do not forget that Questing will have a shelf life of 9 months. This means to use OpenSSL 3.5 (or later) in an Ubuntu environment, we have no choice but to await April 2026. Those time frames may be okay(ish) for those who are using older (or even obsolete) operating systems. The staff and researchers of such businesses will need to experiment with PQC using other OS.

Red Hat Enterprise Linux (RHEL) and Clones

(Clones include CentOS Stream, AlmaLinux, Rocky Linux)

RHEL's PQC roadmap emphasises hybrid cryptography in libraries and kernel for 2030 compliance, with initial support in RHEL 10. RHEL 10 ships OpenSSL 3.2. RHEL and its clones have a strong future, but similar to Ubuntu, they are a step behind the leading edge distributions.

Last week I created a new AlmaLinux 10 (that uses OpenSSL 3.2) and tried to build OpenSSL 3.5 from source code. It was very messy and created a Frankenstein that did not work. Who knows, you may be smarter and lucky. Let me know if you tried PQC on RHEL or its clones and how.

Other Notable OS : Fedora and SUSE Tumbleweed

I used Fedora in June but have never used Tumbleweed. Research shows both are fit for experimenting with PQC.

I am conscious that I have left out quite a few other operating systems from this analysis as I do not use them at all. However, I have a hint to make your own research more meaningful. Try to search for the relevant packages (OpenSSL or OpenSSH-server) in the latest version of your favourite OS and if they are 3.5 and 10 respectively, you are possibly off to a robust start.

Annex 1 Overview of OpenSSL 3.5 LTS

OpenSSL 3.5 is the latest Long-Term Support (LTS) version, released on April 8, 2025, with support extending until April 8, 2030. This version marks a significant advancement in PQC by integrating implementations of the three initial NIST-standardised PQC algorithms: ML-KEM for key encapsulation, ML-DSA for digital signatures, and SLH-DSA for stateless hash-based digital signatures. These are built into the default provider, enabling quantum-resistant security for applications like TLS, without requiring external libraries. The implementations are designed to align with NIST's Commercial National Security Algorithm (CNSA) 2.0 guidelines for quantum-safe cryptography.

Compatibility with FIPS 203 (ML-KEM)

OpenSSL 3.5 provides support for ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism), as defined in FIPS 203. Key generation uses a 64-byte seed concatenating the 32-byte 'd' and 'z' parameters from the standard, enabling secure key encapsulation resistant to quantum attacks. This compatibility extends to TLSv1.3 for hybrid quantum-safe key agreements, with performance benchmarks showing ML-KEM key generation nearly as efficient as classical alternatives like **Elliptic Curve Diffie-Hellman (ECDH)**.

Compatibility with FIPS 204 (ML-DSA)

The version includes complete compatibility with ML-DSA (Module-Lattice-Based Digital Signature Algorithm) per FIPS 204, supporting all parameter sets for signature generation and verification. Pre-hash encoding is not natively supported but can be handled externally by users. ML-DSA is integrated for use in certificates and TLS, aligning with NIST's quantum-resistant signing requirements.

Compatibility with FIPS 205 (SLH-DSA)

OpenSSL 3.5 implements SLH-DSA (Stateless Hash-Based Digital Signature Algorithm) as specified in FIPS 205, offering key generation, signing, and verification with small keys but larger signatures and slower operations compared to lattice-based alternatives. Like ML-DSA, pre-hash support is user-managed externally. This provides a hash-based PQC option for scenarios prioritising statelessness and long-term security.

Annex 2 Overview of OpenSSH 10.0

OpenSSH 10.0 is the latest version, released on April 9, 2025, as part of the OpenBSD project's ongoing development. This release introduces significant advancements in PQC, focusing on quantum-resistant key exchange to protect against future quantum computing threats. The primary PQC feature is the adoption of hybrid key exchange algorithms, which combine classical and quantum-resistant methods for backward compatibility and enhanced security. These are integrated by default in ssh and sshd, enabling quantum-safe connections without configuration changes. OpenSSH 10.0 aligns with emerging standards for quantum-safe networking, including those from NIST and CNSA 2.0, though full FIPS certification for PQC components is not yet detailed. Performance impacts are minimal, with the new default **Key Exchange (KEX)** being faster than previous hybrids in benchmarks.

Compatibility with FIPS 203 (ML-KEM)

OpenSSH 10.0 provides support for ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism) as defined in FIPS 203, integrated via the hybrid key exchange method mlkem768x25519-sha256, which is now the default for ssh(1) key agreements. This hybrid combines ML-KEM-768 with Curve25519-sha256, offering quantum resistance while maintaining compatibility with non-PQC clients. It uses a formally verified implementation of ML-KEM for reliability. Support extends to server configurations, with options to customise KEXAlgorithms for specific needs.

Compatibility with FIPS 204 (ML-DSA)

OpenSSH 10.0 does not include native support for ML-DSA (Module-Lattice-Based Digital Signature Algorithm) as specified in FIPS 204. PQC efforts in this release are limited to key exchange mechanisms, with no integration of quantum-resistant digital signatures for host keys or user authentication. Existing signature algorithms remain classical (e.g., Ed25519, RSA), and future versions may add ML-DSA support as NIST standards mature.

Compatibility with FIPS 205 (SLH-DSA)

OpenSSH 10.0 lacks support for SLH-DSA (Stateless Hash-Based Digital Signature Algorithm) per FIPS 205. As with ML-DSA, the focus is solely on key exchange, not signatures, so no hash-based PQC options are available for authentication or certificates. Users relying on hash-based signatures for long-term security may need external tools or future updates.

Annex 3 Overview of Linux Kernel 6.16-rc7

Linux Kernel 6.16-rc7 was released on July 20, 2025. Cryptography-related changes in the broader 6.16 merge window include support for hardware-wrapped encryption keys in fscrypt (for secure inline encryption) and the removal of the CRYPTO_POLY1305 algorithm, but these are classical crypto enhancements rather than post-quantum (PQ) specific.

Post-Quantum Cryptography Features in 6.16-rc7

Research across kernel changelogs, mailing lists, and feature summaries reveals no integrated PQC features in Linux Kernel 6.16-rc7. There are no implementations of NIST-standardised PQC algorithms such as ML-KEM (FIPS 203), ML-DSA (FIPS 204), or SLH-DSA (FIPS 205), nor any hybrid modes combining classical and PQ methods in the crypto API, module signing, IPsec, WireGuard, or filesystems like dm-crypt/fscrypt. The crypto subsystem updates are limited to non-PQ items, such as the aforementioned hardware-wrapped keys for better secure key management and the deprecation of legacy algorithms like POLY1305.

Community discussions, including a DebConf25 talk on rethinking kernel cryptography for the PQ era, highlight the need for future integration but confirm no such patches have landed in 6.16. The talk emphasises preparation steps like aligning with userspace (e.g., OpenSSH/OpenSSL hybrids like Kyber + X25519), updating kernel module signing (replacing RSA/ECDSA), securing VPNs (e.g., IPsec/WireGuard with ML-KEM), and enhancing storage encryption for long-term security, but these remain proposals without mainline commits.

Annex 4 Overview of PQC in other major kernels

PQC in Windows NT Kernel

The Windows NT kernel integrates PQC through its Cryptography API: Next Generation (CNG) and SymCrypt library, which operate in kernel mode for system-level operations like secure boot and device encryption. Windows 11 Insider builds may support FIPS 203/204/205. But I do not use NT Kernel and cannot confirm. If available for kernel-mode providers in BCrypt, enabling quantum-resistant TLS, IPsec, and BitLocker encryption stands to benefit.

PQC in macOS XNU Kernel

The XNU kernel in macOS (e.g., macOS 15 Sequoia betas from June 2025) may incorporate PQC via its kernel-integrated crypto modules and Security framework, focusing on system-wide protections. I am not an iMac user and this para is based on what I could read. Suggest check the latest on the Apple security website to which I have included a link at the end.

References

References

The following sources provide detailed information on post-quantum cryptography standards, software, and operating systems.

1. NIST. (2024, August 13). NIST Releases First 3 Finalised Post-Quantum Encryption Standards. https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards
2. Open Quantum Safe. (2023, March 14). OpenSSL 3 Provider for Post-Quantum Algorithms. https://github.com/open-quantum-safe/oqs-provider
3. OpenSSH. (2024, August 15). OpenSSH 9.8 Release Notes. https://www.openssh.com/txt/release-9.8
4. NIST. (2024, August 13). FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf
5. NIST. (2024, August 13). FIPS 204: Module-Lattice-Based Digital Signature Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf
6. NIST. (2024, August 13). FIPS 205: Stateless Hash-Based Digital Signature Standard. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf
7. Open Quantum Safe. (2023, March 14). OpenSSL 3 Provider for Post-Quantum Algorithms. https://github.com/open-quantum-safe/oqs-provider
8. OpenSSH. (2024, August 15). OpenSSH 9.8 Release Notes. https://www.openssh.com/txt/release-9.8
9. Linux Kernel. (2024, July 14). Linux 6.10 Release Announcement. https://lore.kernel.org/lkml/20240714161204.406108-1-torvalds@linux-foundation.org/
10. Debian. (2025, July). Debian Testing (Trixie) Package: OpenSSL. https://packages.debian.org/trixie/openssl
11. Debian. (2025, July). Debian Testing (Trixie) Package: OpenSSH. https://packages.debian.org/trixie/openssh-server
12. Arch Linux. (2024). Arch Linux Package: OpenSSL. https://archlinux.org/packages/core/x86_64/openssl/
13. Arch Linux. (2024). Arch Linux Package: OpenSSH. https://archlinux.org/packages/core/x86_64/openssh/
14. Red Hat. (2024). Interoperability of RHEL Post-Quantum Cryptography. https://access.redhat.com/articles/7047934

15. Cryptomathic. (2025, June 2). Quantum-Ready Cryptography with OpenSSL 3.5 on RHEL 9.6. https://www.cryptomathic.com/news-events/blog/quantum-ready-cryptography-with-openssl-3.5-on-rhel-9.6

16. AlmaLinux. (2023, September 19). AlmaLinux 9.2 FIPS Compliance. https://wiki.almalinux.org/release-notes/9.2.html

17. Microsoft. (2024, September 9). Microsoft's Quantum-Resistant Cryptography. https://techcommunity.microsoft.com/t5/security-compliance-and-identity/microsoft-s-quantum-resistant-cryptography-is-here/ba-p/4285580

18. Apple. (2024). macOS Security Features. https://developer.apple.com/documentation/security