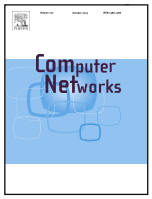
Computer Networks 275 (2026) 111957

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Comparative analysis of post-quantum handshake performance in QUIC and TLS protocols

José A. Montenegro [a,*], Ruben Rios [a], Javier Bonilla [b]

[a] Network, Information and Computer Security (NICS) lab, Universidad de Malaga, Malaga, 29071, Spain
[b] ETSI Informática, Universidad de Malaga, Malaga, 29071, Spain

ARTICLE INFO

Keywords:
Web security
Post-quantum cryptography (PQC)
Transport layer security (TLS)
Quick UDP internet connections (QUIC)
Protocol evaluation

ABSTRACT

The rapid advancement of quantum computing threatens the security foundations of today's communication protocols. On the web, the traditional approach to securing web traffic has been to couple HTTP with TLS over TCP, but QUIC over UDP has recently emerged as an alternative to reduce latency and improve performance over unreliable networks. The need for an urgent transition to quantum-resistant web protocols demands further examination of both stacks. Consequently, in this work we evaluate their performance when combined with post-quantum primitives. To this end, we devise a practical evaluation framework that integrates the cryptographic primitives into TLS and QUIC implementations. Our analysis focuses on comparing the impact of hybrid and post-quantum primitives on TLS and QUIC under both ideal and realistic network conditions, providing quantitative insights into the performance cost and feasibility of transitioning toward a post-quantum web. Our results indicate that hybrid KEMs incur the highest handshake latency and bandwidth overhead, while pure post-quantum KEMs offer a favorable trade-off between security and performance, with only moderate costs. Moreover, QUIC consistently reduces the performance penalty of post-quantum primitives compared to TLS, especially in lossy network environments.

1. Introduction

More than half of the data transmitted over the Internet today relies on the Hypertext Transfer Protocol (HTTP) [1]. Traditionally, HTTP traffic has been carried over the Transmission Control Protocol (TCP), which provides reliable data transport. However, Quick UDP Internet Connections (QUIC) was developed as a transport protocol designed to overcome several limitations of TCP. Leveraging these capabilities, HTTP/3 was introduced as the first version of HTTP to operate over QUIC. Currently, more than 30 % of HTTP traffic is transported over QUIC [2].

The Hypertext Transfer Protocol Secure (HTTPS) is the secure version of HTTP. As of 2025, over 88 % of websites use HTTPS by default [3], reflecting the commitment to web security. In traditional stacks, HTTPS is implemented by layering HTTP over TLS and TCP, where TLS ensures confidentiality and integrity over a reliable transport. In more recent configurations, where HTTP is stacked over QUIC and UDP, TLS is directly integrated into the transport layer . This tight integration is expected to reduce handshake latency and enable novel features such as 0-RTT connections, which are particularly beneficial for performance over lossy or high-latency networks.

Despite known attacks on earlier versions of TLS [4], web security continues to rely on TLS 1.3 for key establishment and authentication. While the cryptographic primitives it employs are considered secure against classical attackers [5], the rapid progress in quantum computing poses a serious threat to the cryptographic foundations of TLS. In the near future, a cryptographically relevant quantum computer – one capable of breaking current cryptographic systems – may become a reality [6,7], making a timely migration to quantum-resistant cryptographic algorithms imperative.

The integration of quantum-resistant (post-quantum) cryptography into security protocols should be pursued without compromising performance or significantly increasing network overhead. Post-quantum algorithms are typically costly and exhibit larger key sizes, ciphertexts and signatures than their classical counterparts, potentially increasing latency and bandwidth usage. Evaluating and understanding how the integration of these algorithms into TLS and QUIC affects overall performance is essential for a smooth transition to a quantum-safe web.

In this work, we present a systematic performance comparison between post-quantum versions of TLS and QUIC that integrate NIST-standardized post-quantum primitives, assuming their underlying cryptographic security assurances provided by the NIST PQC standardization

* Corresponding author.
E-mail addresses: jmmontes@uma.es (J.A. Montenegro), ruben.rdp@uma.es (R. Rios), jbonillac@uma.es (J. Bonilla).

https://doi.org/10.1016/j.comnet.2025.111957
Received 12 June 2025; Received in revised form 11 November 2025; Accepted 17 December 2025
Available online 22 December 2025

process. We devise an evaluation framework to assess two key metrics: handshake latency and packet exchange volume. Our study is structured across three security levels. At each level, we assess traditional, hybrid and purely post-quantum Key Encapsulation Mechanisms (KEMs) with a fixed signature algorithm. The primary goal of this approach is to quantify the performance penalty of switching between KEMs within or across security levels as well as when moving from classical to hybrid or post-quantum.

Additionally, TLS and QUIC are evaluated under two scenarios: ideal and realistic networks with packet loss and delays. This approach enables identifying both the most favorable and the most adverse configurations for each protocol, revealing how design choices – such as QUIC's integrated transport-layer framing versus TLS's layered architecture – interact with emerging post-quantum primitives and network conditions.

The rest of the paper is organized as follows. Section 2 reviews the TLS and QUIC handshake flows and details where cryptographic operations occur. Section 3 analyzes related work and positions our main contributions. Section 4 describes our evaluation setup. Results and their analysis are presented in Section 5, first under ideal setting, followed by realistic network conditions. Finally, Section 6 summarizes our findings and outlines future research directions.

2. Background

This section provides background information on the operation of the two protocols analyzed in this paper, with a focus on the cryptographic operations and the messages involved in the handshake phase. It also provides an overview of the cryptographic primitives involved at this stage.

2.1. Protocols overview

Until recently, TCP was the only transport protocol used to carry HTTPS traffic. One of the main reasons for this is that TCP ensures an ordered and lossless delivery of data, which was not available in the User Datagram Protocol (UDP) and this is crucial for the operation of TLS. However, the development of the QUIC protocol introduced some interesting features that enabled its use for secure web communication.

QUIC operates over UDP, a connectionless protocol that allows datagram transmission without requiring a prior handshake. To compensate for UDP's lack of reliability, QUIC implements its own mechanisms for loss recovery and ordered delivery. This design enables QUIC to implement its own connection establishment mechanism, which is tightly integrated with the TLS 1.3 handshake. By combining transport and cryptographic handshakes, QUIC enables secure communication to be established with fewer round-trip times compared to the traditional TCP + TLS stack. This behavior is illustrated in Fig. 1.

At the transport level, each protocol defines its own header. TCP typically includes a 20-byte header, while UDP adds only 8 bytes. QUIC, built on top of UDP, introduces a variable-length header that depends on the packet type. During the handshake phase, illustrated in Fig. 1(b), all packet types except `1-RTT` use a long header, with a minimum size of 15 bytes, often extended with additional fields depending on the specific packet type. Moreover, each QUIC packet may encapsulate multiple frames-units of data or control information-each preceded by its own frame-specific header, typically starting at 1 byte. However, this behavior is characteristic of the data transmission phase and has no impact on the handshake process.

The QUIC handshake begins with an `Initial` packet sent by the client to the server. This packet contains a TLS `ClientHello` message encapsulated in a `CRYPTO` frame.[1] In response, the server sends `Initial` and `Handshake` packets containing the `ServerHello`, `EncryptedExtensions`, `Certificate`, `CertificateVerify` and `Finished` messages, as defined in the TLS handshake (see Fig. 1(a)).

At this point, with keys already established, encrypted data can start flowing using `1-RTT` packets. The client acknowledges the keys and finalizes the handshake with a `Handshake Done` frame. From this moment, all communication is encrypted and authenticated. However, clients may send `0-RTT` data early by reusing prior session parameters, reducing latency at the expense of some security guarantees.

We next focus on the cryptographic handshake phase, as this is where post-quantum cryptography will be employed to protect against quantum attacks targeting traditional public-key algorithms. In contrast, the data transmission phase relies on symmetric-key cryptography, which is not significantly impacted by such attacks.

2.2. Handshake phase

The cryptographic handshake consists of three stages: the key exchange phase, the server parameters phase and the authentication phase. This is common to both traditional TLS and QUIC + TLS protocol stacks.

During the *key exchange phase*, the client and the server negotiate the cryptographic algorithms and exchange the key material used to protect subsequent transmissions. This is achieved using a key encapsulation mechanism (KEM), which typically consists of three phases [8], as illustrated in Fig. 1(a):

1. **Key generation**: the client uses a `KeyGen()` algorithm to create a key pair (pk, sk). The public key pk is sent to the server using a `ClientHello` message.
2. **Key encapsulation**: the server uses the client's pk to run the `Encaps()` algorithm, which produces a ciphertext ct and a shared secret ss. The server sends ct back to the client within the `ServerHello` message.
3. **Key decapsulation**: The client uses its secret key sk along with the received ciphertext ct to run the `Decaps()` algorithm and recover the shared secret ss.

The `ClientHello` and `ServerHello` messages are used not only for key encapsulation but also to negotiate the list of cryptographic algorithms to be employed during both the handshake and data transmission phases. The cryptographic algorithms used during the handshake phase may include traditional, hybrid or post-quantum algorithms.

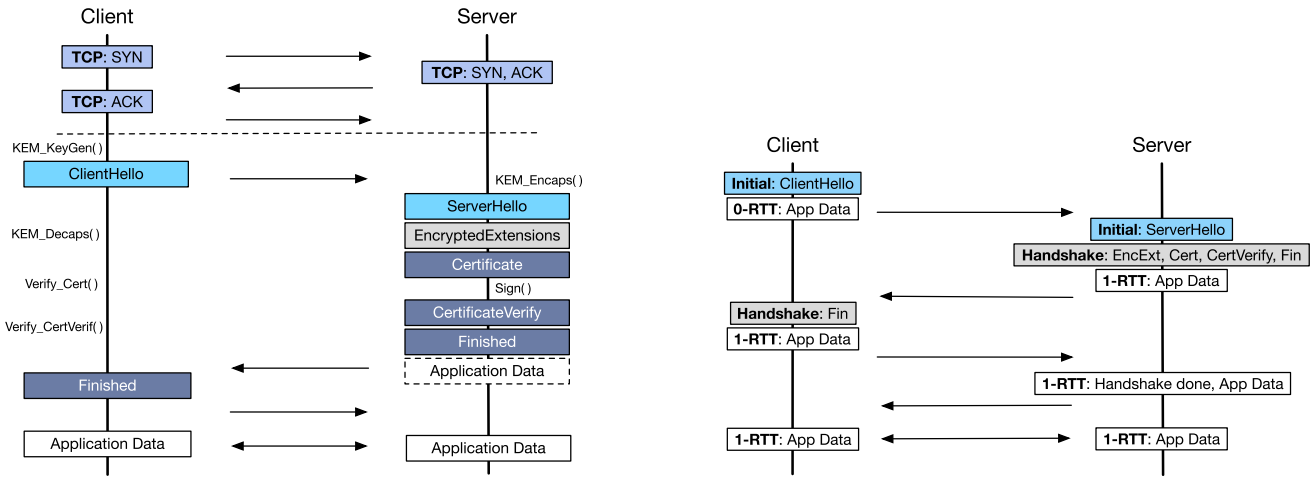
The *server parameters phase* consists of the server transmitting the `EncryptedExtensions` message to the client. This message is sent after key agreement, as it may include sensitive extension data that requires protection. The server may optionally send a `CertificateRequest` message if client authentication is required. However, mutual authentication is uncommon in typical web scenarios and is therefore not represented in Fig. 1(a).

During the *authentication phase*, the server proves its identity to the client, typically by presenting a digital certificate carried in the `Certificate` message. In both TLS and QUIC, the authenticity of the public key is ensured through X.509 digital certificates issued by a Certification Authority (CA) as part of the Public Key Infrastructure (PKI). This infrastructure manages key issuance, validation, and revocation, complementing the cryptographic use of the private key during the handshake process. Proper management of TLS server certificates is critical to maintaining trust and ensuring the integrity of the authentication phase, as highlighted in NIST SP 1800-16 [9].

The verification process to demonstrate possession of the certificate's corresponding private key involves two cryptographic operations:

1. **Signature**: the server generates a digital signature σ of the entire handshake using the `Sign()` algorithm with the private key sk_c associated with its certificate. The signature σ is sent to the client using a `CertificateVerify` message.

[1] The payload of the UDP datagram carrying this packet must be of at least 1200 bytes; otherwise, it is discarded by the server.



(a) TLS 1.3 (based on RFC 8446) (b) QUIC (based on RFC 9000)

Fig. 1. TLS and QUIC handshakes.

2. **Verification**: the client performs two checks. First, it verifies the authenticity of the certificate. Then, it uses the public key of the certificate pk_c to verify the authenticity of the signature σ.

The `Certificate` and `CertificateVerify` messages contain signatures generated using traditional, hybrid or post-quantum signature algorithms.

Finally, the `Finished` messages are exchanged to confirm the integrity of the handshake and to verify that both parties have derived the same shared secret. Upon successful verification, the data transmission phase begins, during which application-level data is encrypted using authenticated encryption with keys derived from the established secret.

2.3. Cryptographic primitives

The primitives involved in the handshake can be classified as traditional, post-quantum or hybrid. Next we provide an overview and examples of KEM and signature primitives of each category.

Traditional primitives:. are cryptographic algorithms based on mathematical problems that are intractable for classical computers but vulnerable to quantum attacks. Notable examples include schemes relying on the integer factorization (IF) and discrete logarithm (DL) problems, both of which can be efficiently solved on a large-scale quantum computer using Shor's algorithm [10] and its subsequent improvements [11].

Common key exchange primitives are Diffie-Hellman (DH), its elliptic-curve variant (ECDH) and more modern constructions defined over Montgomery curves (XDH), designed for improved performance and security. All these schemes are based on the discrete logarithm (DL) problem. Another widely used primitive is the Rivest-Shamir-Adleman (RSA) algorithm, which is based on the integer factorization (IF) problem. Their security is measured in terms of *bits of security*[2] and depends on the choice of parameters size, as shown in Table 1.

Traditional signature algorithms are based on the same underlying hard problems. Examples include RSASA-PSS (an RSA variant optimized for signatures) and the Digital Signature Algorithm (DSA), along with its elliptic-curve counterpart, ECDSA. Modern implementations often adopt Edwards-curve variants, which offer improved performance while maintaining strong security guarantees. Table 1 also includes examples of these primitives.

[2] A security level of n bits implies that the best known attack would require approximately 2^n operations to break the system.

Table 1
Examples of traditional KEM and signature primitives.

	Primitive	Problem	Parameter	Parameter (bits)	Security (bits)
	RSA-2048	IF	Finite field	2048	112
	DH-3072	DL		3072	128
KEM	ECDH-P256			256	128
	x25519			255	128
	ECDH-P384	ECDL	Elliptic curve	384	192
	x448			448	224
	ECDH-P521			521	256
	RSASSA-PSS	IF	Finite field	2048	112
	DSA-2048	DL		2048	112
Signature	ECDSA-P256			256	128
	Ed25519			255	128
	ECDSA-P384	ECDL	Elliptic curve	384	192
	Ed448			448	224
	ECDSA-P521			521	256

Post-quantum primitives:. these primitives are based on mathematical problems or constructions for which no efficient classical or quantum algorithms are currently known [12]. Since quantum computers do not offer a significant advantage over classical ones, these primitives are referred to as post-quantum or quantum-resistant.

Currently, numerous post-quantum cryptographic primitives exist, often classified into various families, including lattice-based, hash-based, and code-based cryptography. Lattice-based primitives include the "ML" family (*Module-Lattice*), such as ML-KEM and ML-DSA, along with the FN-DSA scheme (*Fast Fourier transform over NTRU-Lattice-based Digital Signature Algorithm*). Hash-based primitives are represented by the SLH-DSA standard (*Stateless Hash-based Digital Signature Algorithm*, formerly known as SPHINCS+). Finally, code-based schemes currently include the HQC (*Hamming Quasi-Cyclic*) algorithm.

After several rounds of evaluation organized by the U.S. National Institute of Standards and Technology (NIST), several of these primitives have been standardized or are in the process of being standardized [13]. These algorithms can be parameterized to achieve different security levels, as summarized in Table 2.

The security levels defined by NIST for post-quantum cryptography are related to the amount of resources (classical or quantum) required to break a post-quantum primitive. These levels approximately correspond to the computational effort required for classical attacks, where Levels I, III, and V correspond to the effort needed to break 128-bit, 192-bit,

Table 2
NIST post-quantum KEM and signature primitives (†Two variants exist: small and fast).

	Primitive	FIPS Standard	Family	Parameter Set	Security Level
KEM	ML-KEM (Kyber)	FIPS 203	Lattice-based	ML-KEM512 ML-KEM768 ML-KEM1024	I III V
	HQC	In process	Code-based	HQC-128 HQC-192 HQC-256	I III V
Signature	ML-DSA (Dilithium)	FIPS 204	Lattice-based	ML-DSA44 ML-DSA65 ML-DSA87	II III V
	FN-DSA (Falcon)	In process	Lattice-based	FN-DSA512 FN-DSA1024	I V
	SLH-DSA (SPHINCS$^+$)	FIPS 205	Hash-based	SLH-DSA128† SLH-DSA192† SLH-DSA256†	I III V

Table 3
Summary of NIST security levels and their approximate classical strength.

NIST Level	Classical Security (bits)
Level I	128-bit
Level III	192-bit
Level V	256-bit

and 256-bit symmetric cryptosystems, respectively, as summarized in Table 3.

The early adoption of quantum-resistant primitives mitigates the so-called *Harvest Now, Decrypt Later (HNDL)* threat [14], in which encrypted information intercepted today could be stored and decrypted in the future once quantum computers become practical. While these algorithms are considered quantum-resistant their actual implementations may introduce security issues [15].

Hybrid primitives:. these constructions combine traditional cryptographic primitives with post-quantum ones to provide security against both classical and quantum adversaries. The goal is to ensure that even if one of the components is broken (e.g., by a future quantum computer), the overall system remains secure, leveraging the strengths of both approaches during the transition to post-quantum cryptography. While they provide additional security, they also introduce additional overhead.

3. Related work

This section reviews previous studies focused on the evaluation of protocols that incorporate post-quantum cryptography to secure web communications. In particular, we compare our work with studies that have examined TLS, QUIC, or both.

Several studies have examined the performance of the TLS handshake when integrating post-quantum primitives (see Table 4 for a summary). Notably, Döring and Geitz [16] and [17] analyze various key encapsulation and authentication mechanisms involving both traditional and post-quantum algorithms. Both evaluations are conducted under ideal network conditions. The key difference is that [17] considers standardized primitives and not only evaluates the number of connections but also assesses traffic volume, which becomes particularly relevant when using post-quantum signatures at higher security levels.

While the QUIC protocol has been extensively studied, its integration with post-quantum cryptography has received comparatively less attention, although some studies have begun to address this area. Kempf et al. [18] integrates post-quantum cryptographic protocols in three different QUIC protocol implementations. They evaluate the duration of QUIC handshake as well as the number of packets transmitted for different KEMs and signature algorithms. Their tests were performed under ideal networks conditions, with varying MTU sizes to assess the impact of packet fragmentation. Packet loss and delay are not considered.

Some studies, including ours, focus on comparing the impact of post-quantum cryptography on TLS and QUIC. The paper [19] specifically addresses post-quantum authentication, evaluating handshake duration with two post-quantum signatures under ideal conditions and varying packet loss rates. However, this work does not consider the impact of packet delay or assess traffic volume. Raavi et al. [20] extends this analysis by evaluating the same post-quantum primitives under real network conditions, using QUIC and TLS servers distributed across different geographic locations. In addition, they perform an analysis of the number of packets exchanged and the maximum packet size observed in handshakes.

In general, our paper advances the state of the art by providing a configurable framework that facilitates the evaluation of QUIC and TLS handshake performance in terms of duration and traffic volume. The handshake is evaluated under both ideal and realistic network conditions, incorporating various packet delay and loss models. The evaluation encompasses traditional, post-quantum and hybrid key exchange algorithms, whereas previous works concentrate on pure post-quantum primitives and overlook hybrid schemes. Additionally, this work provides a detailed statistical analysis and comparative evaluation of the results.

4. Evaluation setup

This section outlines the setup employed for the performance evaluation of TLS and QUIC handshakes performed in Section 5. It begins by describing the design of our evaluation framework, followed by a brief explanation and justification of the statistical metrics used to validate the results. Lastly, it introduces the KEM and signature primitives selected for evaluation.

4.1. Evaluation framework

The designed evaluation framework comprises a set of tools and libraries that enable a reproducible and user-friendly assessment of QUIC and TLS protocols. The framework also facilitates their analysis and contributes to a smoother transition toward a quantum-safe web.

As illustrated in Fig. 2, the tools and libraries used in the framework are organized in different components, each serving a different purpose. These components are grouped into a container environment to enable a seamless deployment of client and server machines regardless of the underlying platform architecture. Additionally, the use of containers provides a working environment that is uniform across all evaluations. `Docker` is the technology chosen for this purpose.

Table 4
Related work summary.

Paper	Protocols	PQC algorithms		Network conditions	Evaluation metrics considered
		KEMs	Signatures		
[16]	TLS	Saber Kyber HQC SIKE	Dilithium Falcon Rainbow SPHINCS+	Ideal	handshakes/second
[17]	TLS	ML-KEM	ML-DSA Falcon SPHINCS+	Ideal	handshakes/second handshake size
[18]	QUIC	Kyber BIKE HQC	Dilithium Falcon SPHINCS+	Ideal fragmentation	handshake duration packets/handshake
[19]	QUIC, TLS	–	Dilithium Falcon	Ideal packet loss	handshake duration
[20]	QUIC, TLS	–	Dilithium Falcon	Real	handshake duration packets/handshake

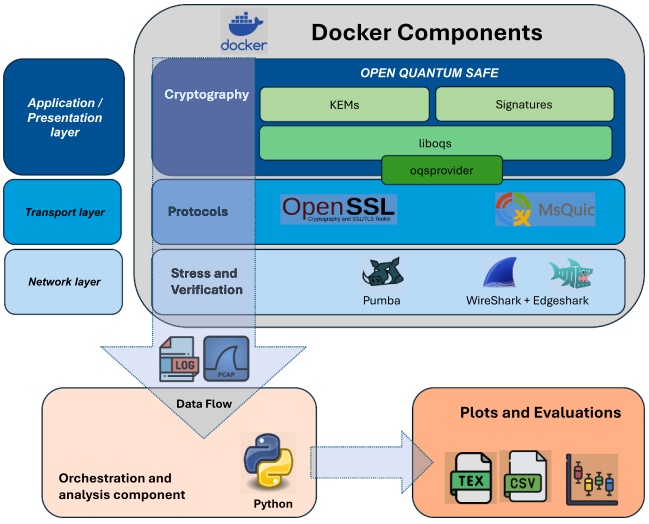


Fig. 2. Evaluation Framework Components and information flows.

Protocols component:. this component includes the protocols to be evaluated, namely TLS and QUIC. In particular, `OpenSSL` and `MsQuic` have been chosen as the specific implementations because of their wide adoption.[3] Both allow the implementation of clients and server enabling the measurement of handshake performance with minimal coding effort.

While `OpenSSL` and `MsQuic` provide implementations for the server, it was necessary to develop client implementations for `MsQuic`. These new clients reproduced the behavior of those offered by `OpenSSL` for measuring TLS handshake performance. Our implementations addressed differences in the way `OpenSSL` and `MsQuic` handle events,[4] which otherwise would result in inaccurate measurements: `OpenSSL` uses blocking functions, whereas `MsQuic` relies on event-driven mechanisms. This adaptation ensures a uniform and consistent evaluation between TLS and QUIC.

Cryptography component:. this component provides the primitives (signatures and KEMs) necessary for the protocols to perform the cryptographic handshake. While both `OpenSSL` and `MsQuic` include traditional signature and KEM primitives, they do not include post-quantum primitives by default. This component relies on the `liboqs` library, from the Open Quantum Safe (OQS) Project, which provides numerous post-quantum cryptographic implementations, including algorithms standardized by NIST as well as proposals from previous standardization rounds.

The integration of these post-quantum primitives into both `OpenSSL` and `MsQuic` is facilitated by their modular architecture, which supports the use of cryptographic providers. The `oqsprovider` acts as a bridging element, allowing both protocol implementations to access the primitives offered by `liboqs`.

Stress and Verification component:. this component enables the emulation of diverse network conditions between the client and the server. This allows handshake performance to be evaluated under both ideal and realistic scenarios, including packet loss and delay.

The emulation of ideal network conditions helps to focus exclusively on the performance of cryptographic primitives during the handshake, avoiding additional variables that could obscure the analysis. However, assessing their behavior under real-world conditions is also essential. To this end, our framework integrates `Pumba`, a flexible and comprehensive tool for manipulating network conditions at the container level.

The Stress and Verification component also employs `EdgeShark`, a tool that enables `Wireshark` to capture packets at the container level. This allows verification that message exchanges between the client and server conform to protocol standards, calculation of the actual network traffic generated by each configuration, and validation of handshake duration measurements against packet transmission timestamps.

Orchestration and analysis component:. this component manages the execution of scenarios and the coordination of all components within the framework through a set of `bash` scripts. The output of executions and captured traffic are processed using `Python` scripts, which produce CSV files with the data of interest. These files are then used to produce plots and statistical analysis of the data for each evaluated scenario. Besides fundamental statistical measures, our scripts evaluate the coefficient of variation, Levene's test and Welch's t-test among others. A concise overview of the statistical measures and their application in our work is provided in the Section 4.2.

4.2. Evaluation metrics

The graphical representation of results is complemented by a thorough statistical analysis, incorporating statistical metrics and significance tests to validate our findings. These are summarized in Table 5.

The `mean` represents the average value of the results and provides a measure of central tendency, while the `standard deviation (std)`

[3] During the course of this work, the latest OpenSSL release introduced a QUIC implementation with both client and server support. However, it was discarded in favor of MsQuic, which was considered a more stable and mature project.

[4] Some procedures in `OpenSSL` are blocking, while `MsQuic` operates using event-driven mechanisms.

Table 5
Statistical metrics and tests.

	Purpose
Mean	Measure central tendency
std	Quantify dispersion around the mean
CV	Variability relative to the mean
IQR	Identify outliers
Shapiro-Wilk	Normality test
Levene's	Homogeneity of variance test
Welch's	Mean difference test

Table 6
Handshake primitives evaluated in TLS and QUIC.

	KEM			
Level	Trad	Hybrid	Post	Signature
I	P-256 x25519	p256_mlkem512 x25519_mlkem512	mlkem512	Ed25519
III	P-384 x448	p384_mlkem768 x448_mlkem768	mlkem768	secp384r1
V	P-521	p521_mlkem1024	mlkem1024	secp521r1

quantifies the dispersion of the results around the mean. Smaller values indicate more stable and predictable results, whereas larger values suggest higher variability. The coefficient of variation (CV) expresses the variability in relative terms, as a percentage of the mean, allowing comparison of variability across results with different scales. Lower CV values indicate more consistent results relative to the mean.

Additionally, the interquartile range (IQR) method is employed to identify unusual results, namely outliers, lying beyond 1.5 times the IQR from the first or third quartile. Counting them gives a sense of how often performance spikes occur.

To ensure the robustness of the comparisons and conclusions, several significance tests are applied. The Shapiro-Wilk test is used to assess whether the data follow a normal distribution. The test returns a p-value; if it is below 0.05, it indicates a significant deviation from normality. This test is a prerequisite for applying other statistical methods, such as t-tests, which assume normal distributed data. Another relevant test used to assess whether multiple groups of results exhibit equal variances is the Levene's test. In case a p-vale below 0.05 is returned it can be concluded that the groups do not present common variance. This check is important because many statistical tests, such as the Welch's t-test, assume that the groups being compared have similar variances. Welch's t-test is used to determine whether the means of two result groups differ significantly.

These metrics and tests enable reliable comparison of handshake performance between any two TLS or QUIC configurations, even in the presence of heavy tails and unequal variability.

4.3. Primitives selection

This work concentrates on KEM algorithms to evaluate the transition to post-quantum TLS and QUIC. The primary reason for concentrating on KEMs is that once a quantum computer is available it could be used to retrieve past keys and decrypt sessions. However, breaking the authentication of previous session is not problematic [21].

The evaluation covers three of the five NIST-defined security levels (cf. Section 2.3) to assess the impact of increasing security requirements on handshake performance. At each level, we include traditional elliptic-curve-based algorithms, hybrid constructions that combine ECC with ML-KEM, and post-quantum ML-KEM variants. The signature algorithms selected for each level are: Ed25519 for Level I, secp384r1 for Level III, and secp521r1 for Level V. A summary of the algorithms evaluated is included in Table 6.

The selected primitives reflect current standards and efficiency considerations: ML-KEM is the only KEM currently standardized by FIPS, and Montgomery elliptic curves are among the most efficient for key exchange. Similarly, Edwards-curve variants are selected for signatures when available, due to their favorable performance characteristics

Table 7
Execution times of KEM operations.

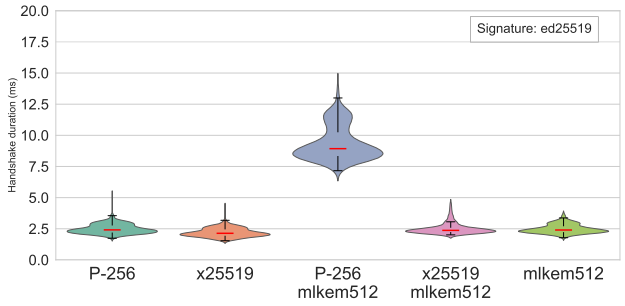
Security Level	KEM Primitives		Operations (ms) KeyGen	Encaps	Decaps	Estimated KEM (ms)
I	P-256	Mean (ms)	0.016	0.112	0.088	0.216
		CV (%)	0	0.8	0.9	
	x25519	Mean (ms)	0.047	0.105	0.053	0.205
		CV (%)	1.1	0.9	0.9	
	p256_mlkem512	Mean (ms)	1.059	0.136	0.415	1.610
		CV (%)	0.47	0.73	1.45	
	x25519_mlkem512	Mean (ms)	0.070	0.124	0.122	0.316
		CV (%)	1.4	1.6	1.6	
	mlkem512	Mean (ms)	0.013	0.012	0.011	0.037
		CV (%)	3.0	2.5	2.7	
III	P-384	Mean (ms)	1.206	2.429	1.222	4.857
		CV (%)	1.2	1.2	2.0	
	x448	Mean (ms)	0.301	0.550	0.243	1.094
		CV (%)	0.7	0.9	0.8	
	p384_mlkem768	Mean (ms)	2.282	2.529	1.637	6.448
		CV (%)	0.5	0.8	0.9	
	x448_mlkem768	Mean (ms)	0.372	0.641	0.639	1.652
		CV (%)	1.1	0.9	0.8	
	mlkem768	Mean (ms)	0.022	0.019	0.019	0.060
		CV (%)	2.8	1.6	4.1	
V	P-521	Mean (ms)	2.810	5.660	2.820	11.290
		CV (%)	1.1	1.2	1.1	
	p521_mlkem1024	Mean (ms)	3.930	5.770	3.270	12.970
		CV (%)	0.8	1.0	0.9	
	mlkem1024	Mean (ms)	0.029	0.027	0.027	0.083
		CV (%)	1.4	1.8	2.2	

5. Performance evaluation

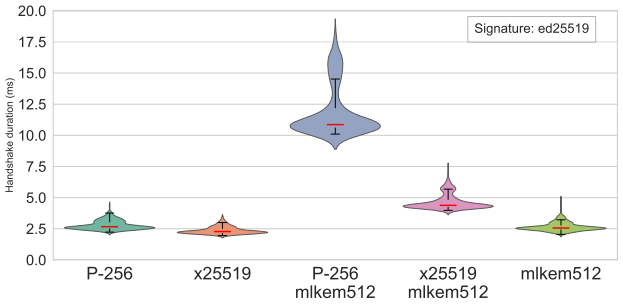
This section presents a detailed evaluation and analysis of the performance of the TLS and QUIC handshakes-as illustrated in Fig. 1-using the selected traditional, post-quantum, and hybrid primitives. The evaluation focuses on both computational and communication overheads under varying network conditions.

The evaluation framework was deployed using Docker, as detailed in Section 4.1, to ensure isolation, reproducibility, and portability. Two configurations are supported: (i) a single-container mode, where both the client and server run within the same Docker instance for local benchmarking; and (ii) a dual-container mode, where the client and server run in separate containers, enabling distributed execution across different hosts. In our evaluation, we employ the second configuration on a single physical server.

The experiments were conducted on a server equipped with an Intel Xeon Silver 4214 CPU with 24 physical cores running at 2.20 GHz. This machine provided a controlled environment to ensure consistent and reliable handshake measurements across all protocols and configurations.



(a) TLS handshake duration at Level I.



(b) QUIC handshake duration at Level I.

	TLS					QUIC				
KEM	Mean	CV	Out.%	Shapiro	Levene	Mean	CV	Out.%	Shapiro	Levene
P-256	2.49	0.15	0.40	7.74e-15		2.79	0.13	1.40	3.47E-17	
x25519	2.20	0.16	0.60	2.26e-12		2.36	0.12	3.80	7.01E-18	
p256_mlkem512	9.36	0.15	1.00	2.8E-17	5.89e-172	11.91	0.17	17.80	4.79E-28	4.14e-106
x25519_mlkem512	2.50	0.17	9.40	1.51E-11		4.64	0.13	9.60	2.41E-18	
mlkem512	2.49	0.14	1.80	1.58E-25		2.65	0.13	6.80	2.41E-18	

(c) Descriptive statistics (Mean, CV, % Outliers) for Level I KEMs under TLS and QUIC.

Fig. 3. TLS and QUIC comparative at Level I.

5.1. Evaluation of KEM operations

During the handshake, both signature and KEM primitives are executed, as illustrated in Fig. 1. Since all configurations within the same security level employ an identical signature algorithm, our analysis isolates the KEM primitives and their respective operations. This approach allows us to clearly quantify their specific contribution to the overall handshake performance, as discussed in the following section.

Table 7 summarizes the execution times (in milliseconds) of the selected KEM primitives for each security level. For each primitive, the table presents the mean duration and coefficient of variation (CV) of fifty executions of each KEM operation. On the client side, both `KeyGen` and `Decaps` are executed, whereas the server performs the `Encaps` operation. To facilitate comparison, an additional column aggregates the three operations, although it should be noted that these are not performed by the same protocol entity.

A straightforward analysis of Table 7 reveals several patterns. As expected, traditional schemes exhibit excellent performance at Security Level I since they rely solely on classical elliptic-curve operations. Hybrid schemes, particularly those based on Edwards curves, show a noticeable increase in computational cost due to the sequential execution of both classical and post-quantum components during key exchange. This performance gap becomes even more pronounced at Security Level III. Finally, the pure post-quantum `ML-KEM` primitives demonstrate remarkably low and consistent execution times across all levels, achieving the lowest total execution times among all evaluated schemes. These results confirm their efficiency and scalability across different parameter sets.

5.2. Evaluation of handshake duration

This section analyzes the time required to complete handshakes for both TLS and QUIC. It also offers a comparative analysis across security levels, using the corresponding KEM primitives for each level. The client and server run in isolated containers under optimal network conditions. The results[5] are based on 500 handshake executions (Fig. 5).

Level I:. At this security level, the highest-performing KEM is traditional `X25519`, with average handshake times of approximately 2.2 ms in TLS and 2.4 ms in QUIC. As evidenced by the comparable average handshake times of `P-256` and `mlkem512`, adopting post-quantum KEM incurs minimal performance overhead. In contrast, hybrid schemes introduce a higher penalty, especially for `p256_mlkem512`, with an average handshake duration of approximately 9 ms in TLS and 12 ms in QUIC. Notably, `x25519_mlkem512` remains competitive in TLS, but nearly doubles handshake duration in QUIC. Complete results are shown in Fig. 3.

The coefficient of variation (CV) reveals that QUIC exhibits lower handshake time variability than TLS, indicating more consistent performance. However, QUIC's higher outlier rate suggests occasional latency spikes, despite its tighter central distribution. This pattern holds across all tested KEMs at this security level, except for `p256_mlkem512`, which shows higher CV in QUIC.

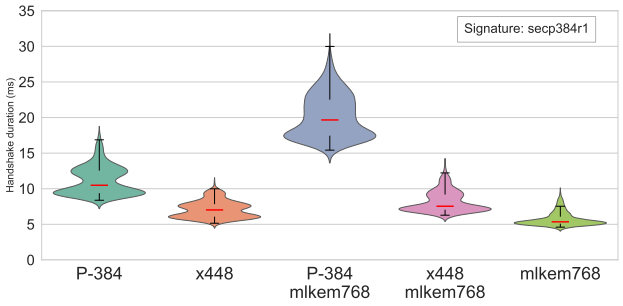
To assess the statistical significance of the observed differences, we performed Welch's t-tests for all KEM pairs within each protocol[6]. Prior to these tests, Shapiro-Wilk and Levene's tests confirmed non-normal distributions and unequal variances across KEMs, respectively, justifying the use of Welch's method (see Fig. 3). The results indicate that choosing among `P-256`, `mlkem512` and `x25519_mlkem512` does not result in a substantial change in the average handshake time in TLS. In contrast, for QUIC, all pairwise KEM comparisons yield statistically significant differences, indicating that KEM selection directly impacts measurable latency.

Level III:. As illustrated in Fig. 4, QUIC consistently outperforms TLS across all KEMs at this security level. The post-quantum `mlkem768` emerges as the most efficient option, with average handshake times just below 4 ms in QUIC and slightly above 5 ms in TLS. While traditional KEMs maintain comparable performance levels, hybrid variants show significantly degraded efficiency - particularly `p384_mlkem768`, which exhibits the most substantial performance penalty.

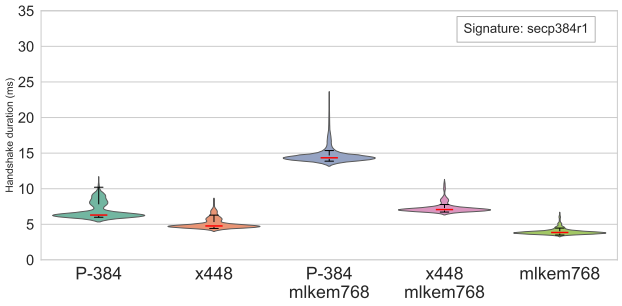
Additionally, `mlkem768` exhibits a low CV coupled with a moderate outlier rate, indicating a stable performance with occasional

[5] A small number of outliers were excluded to improve figure clarity.

[6] Complete test results are omitted for brevity but are available in our GitHub repository: https://github.com/montenegro-montes/TLS-QUIC.



(a) TLS handshake duration at Level III.

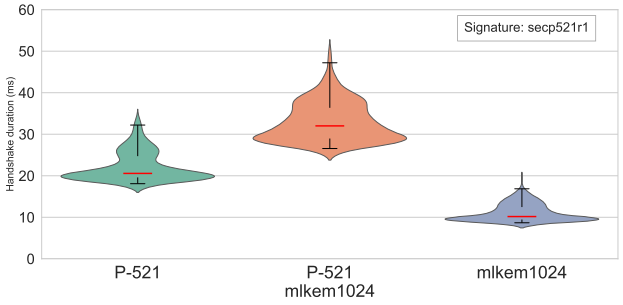


(b) QUIC handshake duration at Level III.

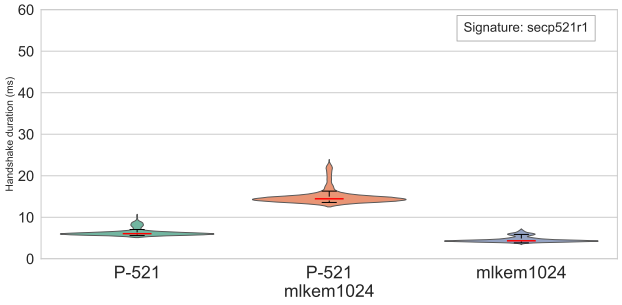
	TLS					QUIC				
KEM	Mean	CV	Out.%	Shapiro	Levene	Mean	CV	Out.%	Shapiro	Levene
P-384	11.12	0.18	0.40	3.41e-17		6.93	0.16	0.60	5.28e-27	
x448	7.09	0.16	0.20	1.11e-23		5.09	0.14	8.20	3.16e-28	
p384_mlkem768	20.14	0.15	0.00	1.03e-13	1.22e-159	14.73	0.08	13.40	9.93e-34	2.59e-22
x448_mlkem768	8.18	0.18	1.20	7.79e-13		7.29	0.09	13.40	4.07e-27	
mlkem768	5.71	0.16	6.40	4.96e-19		3.96	0.11	10.20	6.25e-31	

(c) Descriptive statistics (Mean, CV, % Outliers) for Level III KEMs under TLS and QUIC.

Fig. 4. TLS and QUIC comparative at Level III.



(a) TLS handshake duration at Level V.



(b) QUIC handshake duration at Level V.

	TLS					QUIC				
KEM	Mean	CV	Out.%	Shapiro	Levene	Mean	CV	Out.%	Shapiro	Levene
P-521	22.16	0.16	1.20	2.46e-21		6.33	0.13	13.00	1.55e-28	
p521_mlkem1024	32.96	0.14	0.60	4.37e-15	8.82e-40	14.99	0.12	11.80	2.33e-31	1.52e-15
mlkem1024	11.09	0.19	0.40	2.16e-19		4.67	0.15	11.40	1.52e-15	

(c) Descriptive statistics (Mean, CV, % Outliers) for Level V KEMs under TLS and QUIC.

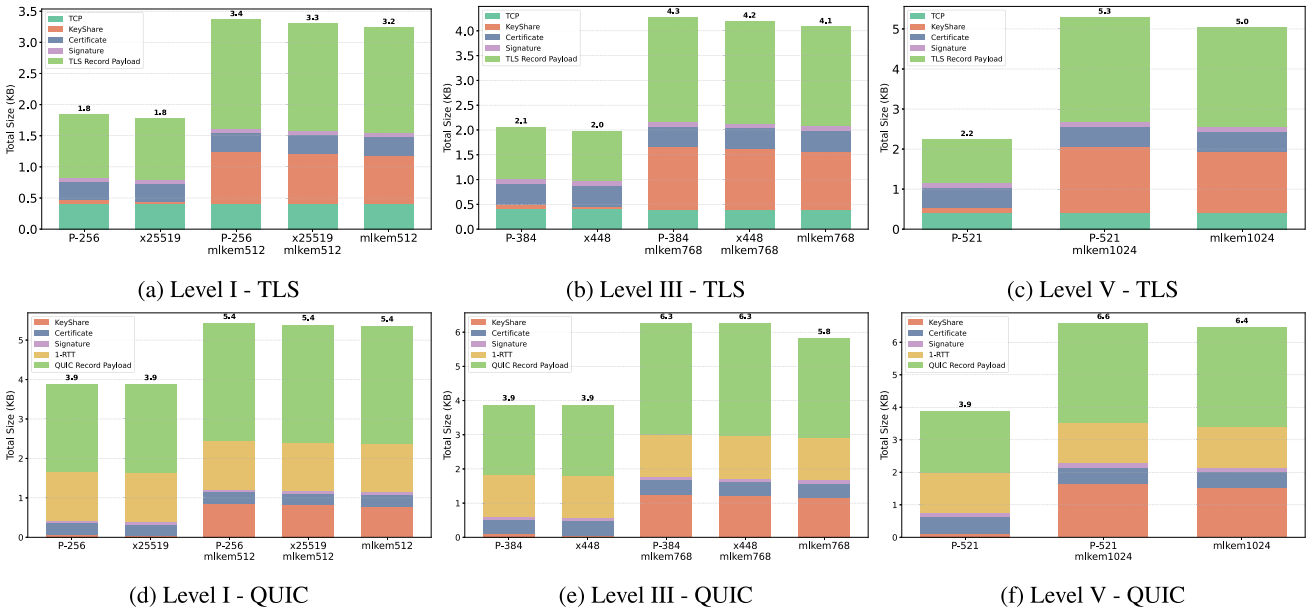
Fig. 5. TLS and QUIC comparative at Level V.

latency spikes. Among traditional KEMs, `P-384` shows the highest overall stability, while `X448` maintains consistent performance in TLS but exhibits more variability in QUIC. Hybrid schemes present an interesting trade-off in QUIC: they achieve very low CVs-suggesting tight clustering around the mean, but suffer from high outlier rates, revealing frequent extreme delays despite their overall consistency.

Welch's t-test at this level demonstrates that every KEM differs significantly from the others under both TLS and QUIC.

Level V:. The results exhibit the same pattern as Level III. The post-quantum scheme `mlkem1024` continues to perform best, with an average of 4.7 ms in QUIC and 11.1 ms in TLS. This is followed by traditional `P-521` with a handshake time of approximately 6.3 ms in QUIC, but almost quadrupling in TLS. The hybrid primitive has the highest latency, 15 ms in QUIC and over 30 ms in TLS.

Notably, although hybrids exhibit slightly lower CVs in QUIC (0.12) than in TLS (0.14), they still suffer from a non-trivial fraction of extreme



(a) Level I - TLS (b) Level III - TLS (c) Level V - TLS

(d) Level I - QUIC (e) Level III - QUIC (f) Level V - QUIC

Fig. 6. TLS and QUIC Handshake traffic size.

Table 8
Certificate, signature and key sizes for the evaluated primitives.

Security Level	KEM Primitives	Operations size (bytes) keyshare	certificate	signature
I	P-256	65	306	64
	x25519	32	306	64
	p256_mlkem512	865	306	64
	x25519_mlkem512	832	306	64
	mlkem512	800	306	64
III	P-384	97	431	103
	x448	56	431	103
	p384_mlkem768	1281	431	103
	x448_mlkem768	1240	431	103
	mlkem768	1184	431	103
V	P-521	133	504	139
	p521_mlkem1024	1701	504	139
	mlkem1024	1568	504	139

delays: more than 11 % of their handshakes in QUIC are classified as outliers. In contrast, the outlier rates in TLS decrease to almost zero, although their CVs increase to 0.14-0.16.

In order to verify these differences, a Welch's t-test showed that all pairwise comparisons were significant in both QUIC and TLS, indicating that no two KEMs share the same mean.

5.3. Handshake traffic size evaluation

This section evaluates and analyzes the total number of bytes exchanged during the handshake phase. We start by compiling Table 8, which summarizes the main cryptographic components contributing to the overall handshake size, including the key share, certificate, and signature fields. These values were obtained by capturing and parsing the TLS and QUIC handshake packets using their corresponding TLS session keys. As expected, traditional schemes exhibit compact parameters, while hybrid and post-quantum KEMs significantly increase the key share size-reaching from approximately 0.8 KB at Security Level I to more than 1.7 KB at Level V. The resulting total handshake sizes for each configuration are illustrated in Fig. 6.

Traditional elliptic-curve schemes required approximately 4 KB in QUIC and 1.8–2.2 KB in TLS, depending on the security level. Pure post-quantum KEMs exchanged roughly 5.4–6.4 KB in QUIC and 3.2–5 KB in TLS. This corresponds to a 45-56 % increase in TLS traffic and a 60–72 % increase in QUIC traffic, compared to traditional schemes.

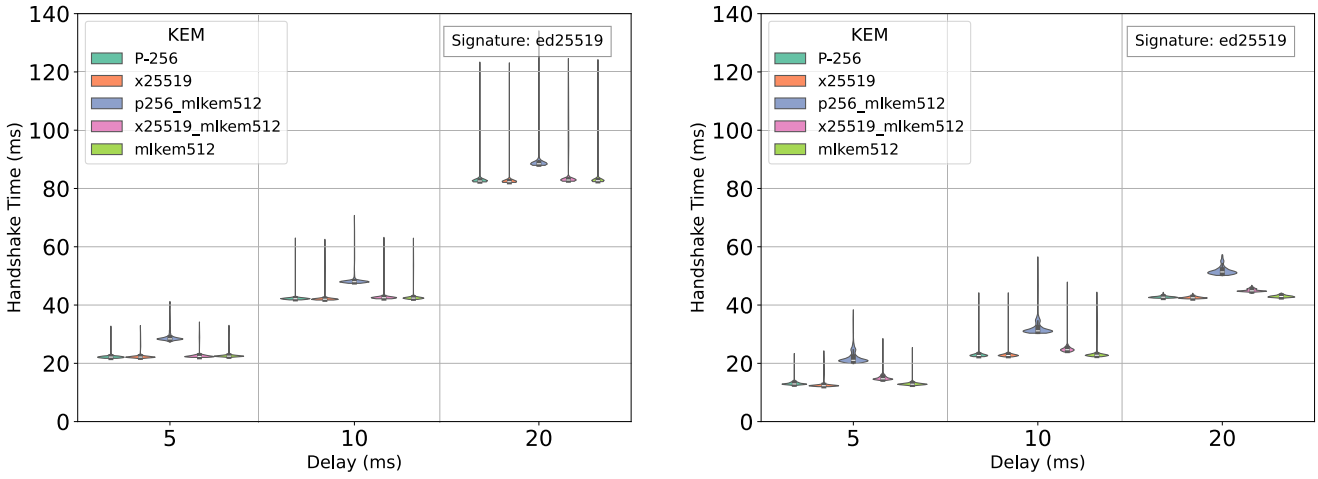
As expected, hybrid KEMs consistently resulted in the highest handshake traffic volumes, exceeding traditional schemes by 2 KB on average and marginally surpassing pure post-quantum KEMs. For example, `p256_mlkem512` exchanged 5.4 KB in QUIC vs. 3.4 KB in TLS; `p384_mlkem768` and `x448_mlkem768` reached 6.3 KB in QUIC and approximately 4.3 KB in TLS; and `p521_mlkem1024` peaked at 6.6 KB in QUIC and 5.3 KB in TLS.

QUIC consistently transmitted more handshake traffic than TLS across all KEMs and security levels, despite avoiding TCP connection establishment. The TLS-to-QUIC traffic ratio ranged from approximately 46 % (Level I, `x25519`) to 81 % (Level V, hybrid). This stems primarily from QUIC's requirement: Initial packets must contain at least a 1.2 KB payload (cf. Section 2.1), inherently increasing the traffic volume.

A systematic evaluation of the bandwidth-security trade-offs between KEM categories and protocols follows. This analysis provides insight for selecting optimal configurations based on specific security requirements and network constraints.

The most significant performance penalty occurs when upgrading from traditional KEMs at Level I to hybrid KEMs at Level V, under TLS this transition incurs an additional handshake latency of approximately 30.6 ms, whereas under QUIC the corresponding latency penalty is only about 12 ms. In contrast, migrating from hybrid, using p-curves, to pure post-quantum KEMs can actually result a latency reduction in QUIC, up to 7 ms when moving Level I to V and 10 ms for Level III to V, accompanied by only modest increases in transmitted bytes. This suggests than an increment of security level in QUIC has not performance penalty in these cases. Averaged across all KEM families and levels, QUIC consistently exhibits significantly lower packet-loss penalties than TLS.

Finally, to assess overall efficiency, we computed the time-per-byte ratio (handshake duration / transmission size) for each configuration. Hybrid KEMs exhibited both the longest handshakes and the highest time-per-byte ratios, confirming their inefficiency. In contrast, post-quantum KEMs maintain a good trade-off between latency and bandwidth. Notably, `mlkem512` in QUIC achieved better ratio than traditional primitives.



(a) TLS handshake duration with delays at Level I.

(b) QUIC handshake duration with delays at Level I.

KEM	TLS							QUIC						
	CV			Outliers %			Slope	CV			Outliers %			Slope
	5	10	20	5	10	20		5	10	20	5	10	20	
P-256	0.03	3.25	1.78	1.8	4.0	7.0	4.44	0.05	0.04	0.05	2.6	3.4	5.0	2.00
x25519	0.03	0.02	1.79	4.2	5.0	6.2	4.37	0.05	0.04	0.05	5.4	2.6	2.6	2.02
p256_mlkem512	0.03	2.90	2.20	8.2	9.4	6.2	4.75	0.08	0.06	0.05	11.2	11.4	9.2	2.03
x25519_mlkem512	0.03	3.22	1.78	1.4	3.6	4.6	4.44	0.06	0.05	0.04	1.8	3.0	0.6	2.03
mlkem512	0.02	3.23	2.33	3.8	6.0	4.6	4.62	0.05	0.04	0.04	7.0	5.0	1.4	2.02

(c) Descriptive statistics (CV, % Outliers) for Level I KEMs under TLS and QUIC with delays.

Fig. 7. TLS and QUIC comparative with delays at Level I.

5.4. Delay sensitivity analysis

To evaluate the impact of network latency on handshake performance, a total of 52 000 handshakes were conducted with simulated delays of 5, 10, and 20 ms in client and server. For each unique combination of protocol, security level, and KEM, a linear regression model is used to extract a slope representing the average handshake time penalty incurred for each millisecond of network delay.

Level I:. At this level, performance ranking is not altered by additional delays in TLS and QUIC (see Fig. 7). Hybrid `p256_mlkem512` exhibits the highest latency, while traditional and pure post-quantum KEMs demonstrate comparable performance. Notably, the transition from ideal network conditions to a 5 ms delay shows `x25519` experiencing the most severe performance degradation, with handshake times increasing by a factor of 10 in TLS and 5.3 in QUIC. Beyond the 5 ms delay threshold, all KEMs exhibit similar degradation, increasing linearly with network delay.

Packet delays amplify variability and spike frequency in both TLS and QUIC, but in distinct ways. For TLS, we observe a non-linear response where the coefficient of variation (CV) remains low (0.3) at 5 ms delay, spikes sharply around 3 at 10 ms-except for `x25519` which remains stable-and settles around 2 at 20 ms delays. Outlier rates follow a similar trend, peaking at intermediate delays. Overall, delays consistently degrade predictability and increases the frequency of extreme delays. Notably, all KEMs exhibit remarkably consistent linear scaling (slope around 4.4 ms/ms) under sustained delays, with `p256_mlkem512` showing the steepest degradation (4.75 ms/ms) and `mlkem512` following closely.

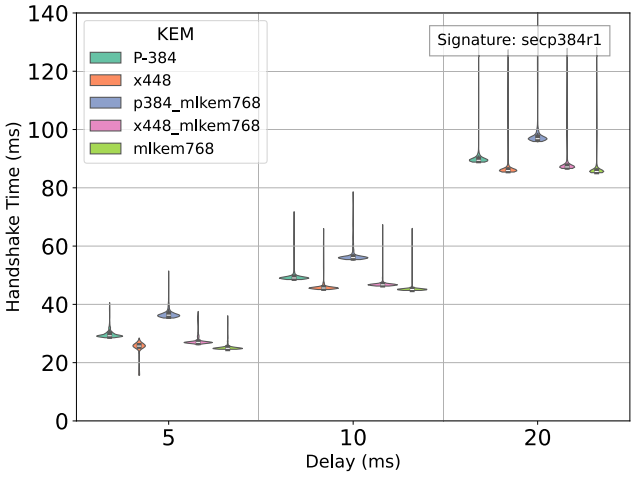
In contrast, QUIC demonstrates significantly more stable performance under increasing network delays. The CV remains consistently low (0.04-0.08) across all delay levels, showing minimal dispersion changes. While QUIC's outlier rates begin higher than TLS's at 5 ms delay, they increase only modestly at 10 ms before stabilizing or even decreasing at 20 ms. This pattern reveals QUIC's ability to maintain a tight performance distribution regardless of network conditions, though it does sustain a slightly larger tail of occasional slow handshakes compared to TLS. Remarkably, the slope is consistent across all KEMs, with an approximate value of 2ms/ms. This represents less than half the value observed in TLS.

Level III:. Consistent with Level I observations, network delays do not affect the relative performance ranking of KEMs at Level III (Fig. 8). Hybrid `p384_mlkem768` exhibits the highest latency, while `mlkem768` maintains the best performance in TLS. At 5 ms delay, the variability (CV) is negligible in most cases; peaks to around 3.0 at 10 ms for the worst-performing configuration; and returns uniformly to approximately 2.0 at 20 ms. Outlier rates fluctuate between 6 % and 20 %.

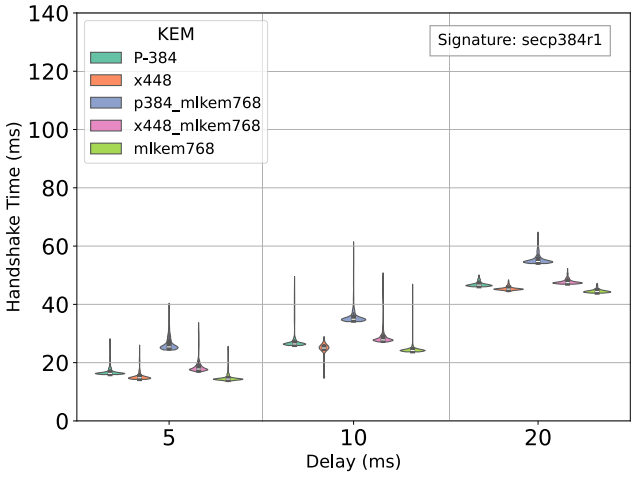
In contrast, the handshake latency increase of QUIC is less pronounced-average slope of 2.0 ms/ms compared to over 4.5 in TLS-from 15 to 26 ms at 5 ms delay to 47-55 ms at 20 ms delays. The CV remains stable below 0.1 at lower delays, though some cases (notably `mlkem768`) experience higher variability at 20 ms delays. Outlier rates peak at moderate delays before stabilizing near 10 % under heavier delays.

Consistent with Level I findings, TLS amplifies network delay into both higher latency and greater variability, whereas QUIC demonstrates superior resilience: it maintains predictable handshake timing and mitigates the impact of network delay.

Level V:. At this security level, `mlkem1024` achieves the lowest mean handshake times, along with the most favorable delay sensitivity



(a) TLS handshake duration with delays at Level III.

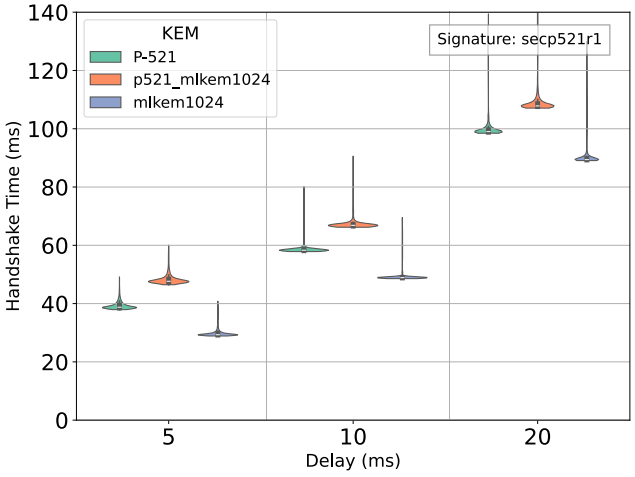


(b) QUIC handshake duration with delays at Level III.

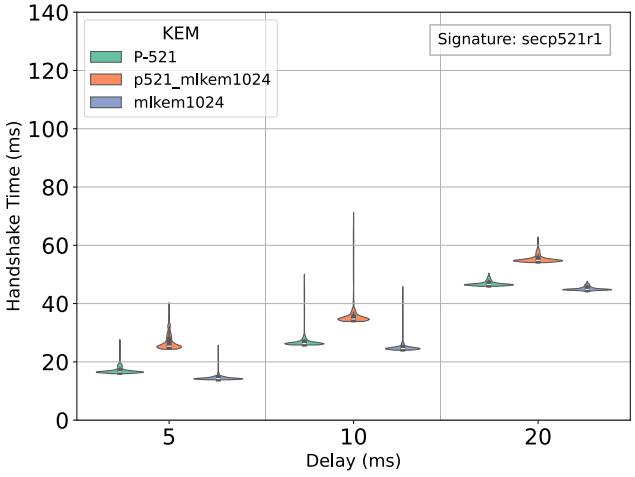
KEM	TLS							QUIC						
	CV			Outliers %			Slope	CV			Outliers %			Slope
	5	10	20	5	10	20		5	10	20	5	10	20	
P-384	0.04	2.84	2.17	7.8	9.0	9.4	4.74	0.05	0.05	2.84	12.6	10.6	9.6	2.00
x448	0.11	0.02	2.25	20.6	9.2	9.6	4.69	0.06	0.13	0.02	8.0	19.6	10.8	2.01
p384_mlkem768	3.66	2.52	2.03	8.8	11.4	12.0	4.46	0.09	0.05	2.52	8.6	12.6	10.2	1.99
x448_mlkem768	0.03	0.02	2.22	12.0	8.2	11.6	4.68	0.07	0.05	0.02	5.0	5.4	8.0	2.02
mlkem768	0.03	3.06	2.26	6.0	4.2	12.2	4.79	0.05	0.05	3.06	10.4	6.2	6.2	2.03

(c) Descriptive statistics (CV, % Outliers) for Level III KEMs under TLS and QUIC with delays.

Fig. 8. TLS and QUIC comparative with delays at Level III.



(a) TLS handshake duration with delays at Level V.



(b) QUIC handshake duration with delays at Level V.

KEM	TLS							QUIC						
	CV			Outliers %			Slope	CV			Outliers %			Slope
	5	10	20	5	10	20		5	10	20	5	10	20	
P-521	3.45	2.44	1.99	8.8	7.2	11.0	4.49	0.06	0.05	0.04	13.6	11.2	10.0	2.02
p521_mlkem1024	2.89	2.16	1.84	10.8	8.6	11.0	4.60	0.09	0.06	0.05	6.6	10.8	12.4	2.00
mlkem1024	0.03	0.02	0.02	9.2	5.2	10.6	3.97	0.05	0.04	0.04	5.2	7.6	9.0	2.04

(c) Descriptive statistics (CV, % Outliers) for Level V KEMs under TLS and QUIC with delays.

Fig. 9. TLS and QUIC comparative with delays at Level V.

Table 9
Variation between ideal case and 20 % loss for each KEM at Level I. Slope is ms increase per 1 % loss.

	TLS					QUIC				
KEM	Δ Mean	Δ CV	Δ Outliers	Δ Size	Slope	Δ Mean	Δ CV	Δ Outliers	Δ Size	Slope
P-256	263.69	2.07	17.6	35	13.40	290.97	2.36	18.1	1065	16.32
x25519	270.30	2.04	18.2	41	17.96	273.52	2.62	14.7	1086	15.58
p256_mlkem512	332.61	1.84	1.4	88	18.83	198.57	1.92	0.7	3338	10.46
x25519_mlkem512	286.67	1.88	10.6	59	15.10	243.51	2.12	9.2	2725	14.12
mlkem512	287.47	1.85	18.2	47	14.11	279.01	1.99	15.3	2517	14.73

(smallest slope). Conversely, `p521_mlkem1024` has the highest performance penalty. The CV for `mlkem1024` remains stable across all tested delays, while other primitives show significantly higher variability. Outlier rates generally cluster around 10 %, though with some notable exceptions. These patterns are detailed in Fig. 9(a).

QUIC exhibits a similar pattern for all configurations, except that KEM slopes are less than half those of TLS and nearly identical across primitives. The CV remains consistently low in all cases. Outlier rates range between 5 and 13 %. Again, `mlkem1024` remains the fastest and most stable across delays.

In summary, network latency exhibits an approximately linear impact on handshake durations in both protocols. QUIC's lower slope (≈ 2 ms/ms) makes it markedly more delay-tolerant than TLS (≈ 4.5 ms/ms). Hybrid KEMs show the greatest sensitivity and variability under delay, while pure post-quantum KEMs occupy an intermediate position between traditional and hybrid schemes.

5.5. Loss sensitivity analysis

The impact of packet loss on handshake performance was evaluated using two complementary approaches. First, uniform packet loss rates of 5 %, 10 %, and 20 % were applied to establish baseline performance degradation. Second, the Gilbert-Elliott model (a two-state Markov chain) is used to replicate correlated loss patterns typical in real networks.

5.5.1. Uniform loss pattern

Each security level is evaluated by comparing the variation in four key metrics between the ideal scenario and one with 20 % packet loss: change in mean handshake time, CV, outlier rate, and average handshake size. Additionally, we incorporate the slope of linear regression (ms increase per 1 % loss) to quantify sensitivity to loss. This set of metrics illustrates the extent to which significant packet loss degrades protocol performance.

Level I:. Under TLS, traditional `P-256` shows the smallest increase in mean handshake time and only moderate variability, while hybrid `p256_mlkem512` incurs the highest latency penalty and greatest variability. Notably, under QUIC, these KEMs exhibit the opposite behavior. In both cases, the post-quantum `mlkem512` falls in between, with performance close to `P-256` while providing resilience to quantum attacks. Detailed results on performance degradation are presented in Table 9.

The regression slopes in TLS (13-19 ms per % loss) and QUIC (10-16 ms per % loss) reveal a near-linear relationship between packet loss and handshake latency, underscoring that KEM choice crucially shapes robustness under adverse network conditions. Most importantly, QUIC consistently incurs a lower latency penalty per % of loss than TLS.

Focusing on handshake size increase, TLS shows only modest growth-limited to a few bytes-indicating minimal retransmission overhead. In contrast, QUIC experiences increases of over 1 KB for traditional KEMs, 2 KB for post-quantum KEMs, and up to 3 KB for `p256_mlkem512`, driven by minimum packet payload size of QUIC's Initial packets. These results indicate that, beyond latency penalties, heavy packet loss can substantially increase bandwidth consumption, particularly during QUIC handshakes.

Level III:. The most robust option in the presence of packet loss both for TLS and QUIC is `mlkem768`. In contrast, `p384_mlkem768` is observed to incur the most significant latency penalty in TLS, despite its negative CV increment and minimal outliers[7] In QUIC, the greatest performance decline is exhibited by `P-384`. Table 10 illustrates the results of our evaluation.

Linear regression slopes remain consistent with the previous security level for TLS (14–19 ms per % loss) demonstrate a consistent degradation. In contrast, QUIC exhibits a more effective adaptation to packet loss, with less pronounced slopes (9–12 ms per % loss), except in the case of using `P-384` where it peaks over 20 ms per %loss.

Handshake size increment in TLS is negligible. However, a more pronounced trend is evident in QUIC, particularly in hybrid primitives, with increments of approximately 5 KB in the worst case.

Level V:. Pure post-quantum `mlkem1024` remains the more resilient to packet loss at this security level, demonstrating the flattest slope both in TLS (≈ 13) and QUIC (≈ 10), as well as the second lowest handshake size increase. In contrast, the hybrid `p521_mlkem1024` suffers the largest latency penalty in TLS, while in QUIC `P-521` exhibits the largest mean variation and CV. Table 11 provides detailed results on the impact of 20 % packet loss compared to the ideal case for Level V.

In conclusion, our cross-level analysis reveals consistent performance patterns. Post-quantum `mlkem` consistently offer the best performance in the presence of uniform packet loss, whereas hybrid schemes incur the heaviest penalties, and traditional schemes sit in between. Generally, QUIC exhibits lower mean delay increase than TLS, but it shows latency spikes and greater bandwidth inflation under loss. These trends underline that the KEM category (traditional, hybrid, or post-quantum) has a predictable impact on performance, while protocol choice (QUIC vs TLS) further modulates the trade-off between speed, variability, and loss resilience.

The analysis of uniform loss rates illustrates how packet loss impacts the handshake, effectively turning a regular handshake into an outlier. Incorporating a model that reflects real-world conditions is considered essential to complete the loss analysis.

5.5.2. Correlated loss pattern

The Gilbert-Elliot model is widely used to simulate real-world packet loss behavior in networks. This model defines two states: a *good* state, where packet delivery is nearly perfect, and a *bad* state, where losses are highly probable. By tuning the loss and transition probabilities between these states, the model replicates bursty loss patterns observed in real networks.

Two scenarios-stable and unstable-are evaluated to compare the resilience of TLS and QUIC under both mild and severe loss conditions. Table 12 summarizes the parameters used in both scenarios: p_g and p_b denote the probability of transitioning from good to bad, and from bad to good, respectively, while ϵ_h and ϵ_k represent the packet loss probability in the bad state and the good state, respectively.

The results are shown in Figs. 10 and 11, corresponding to the stable and unstable scenarios, respectively. All plots use a base-10 logarithmic

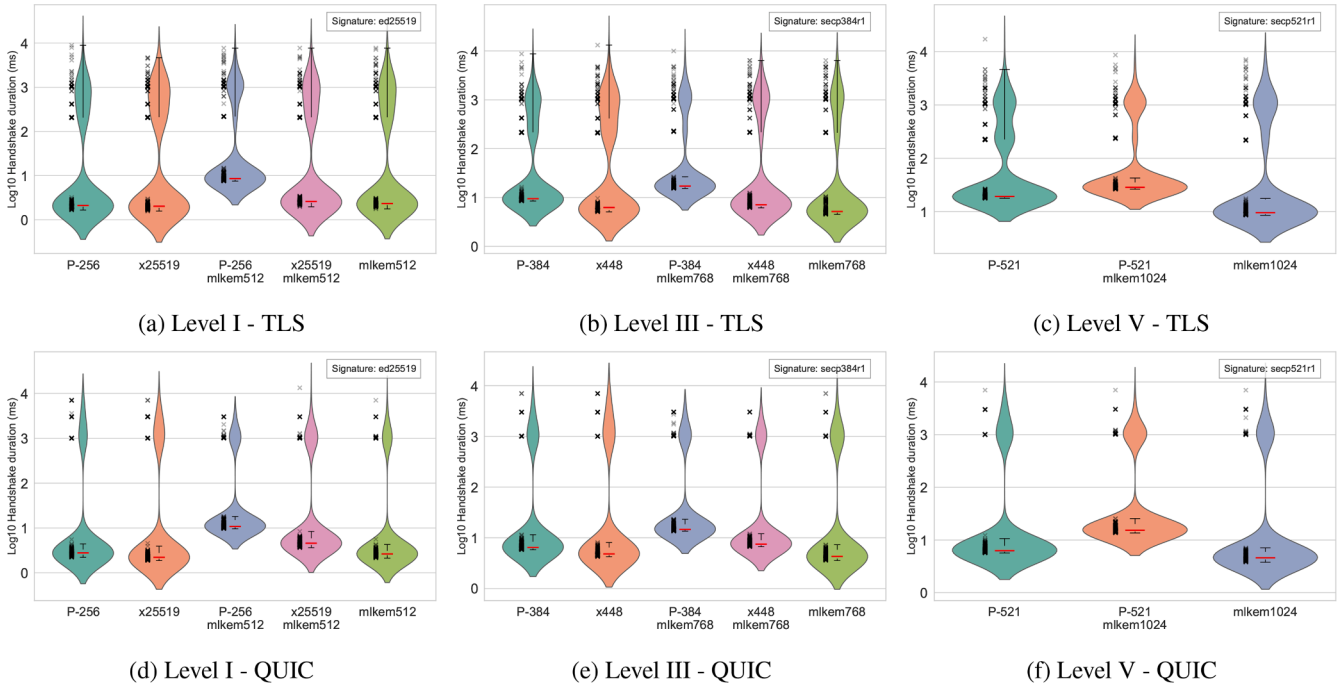
[7] It should be noted that these observations are derived from a high-value scenario in the ideal condition case.

Table 10
Variation between ideal case and 20 % loss for each KEM at Level III. Slope is ms increase per 1 % loss.

	TLS					QUIC				
KEM	Δ Mean	Δ CV	Δ Outliers	Δ Size	Slope	Δ Mean	Δ CV	Δ Outliers	Δ Size	Slope
P-384	280.19	1.79	19.4	47	14.03	363.53	2.77	17.7	958	20.76
x448	304.76	1.78	21.2	46	16.45	232.70	2.73	7.9	1028	11.97
p384_mlkem768	331.93	-3.96	1.8	143	18.67	222.26	1.98	4.7	5094	11.95
x448_mlkem768	271.95	2.04	19.2	85	15.63	195.42	2.07	4.9	3690	9.19
mlkem768	271.35	2.24	18.0	70	14.68	182.10	2.07	8.1	1976	8.96

Table 11
Variation between ideal case and 20 % loss for each KEM at Level V. Slope is ms increase per 1 % loss.

	TLS					QUIC				
KEM	Δ Mean	Δ CV	Δ Outliers	▵ Size	Slope	Δ Mean	Δ CV	Δ Outliers	Δ Size	Slope
P-521	297.34	1.77	20.0	42	14.75	354.43	2.41	7.6	1079	19.65
p521_mlkem1024	300.66	1.98	21.6	192	17.07	238.17	1.85	9.6	4935	11.76
mlkem1024	251.46	2.36	21.8	106	12.71	182.25	2.14	4.9	3674	9.89



(a) Level I - TLS (b) Level III - TLS (c) Level V - TLS

(d) Level I - QUIC (e) Level III - QUIC (f) Level V - QUIC

Fig. 10. QUIC and TLS Handshake duration in a stable network.

Table 12
Gilbert-Elliot parameters for stable and unstable networks.

Scenario	p_g	p_b	ϵ_h	ϵ_k
Stable	0.10	0.50	0.70	0.10
Unstable	0.20	0.40	0.90	0.20

y-axis to highlight the dramatic latency increase compared to the ideal, lossless scenario. Their analysis is organized by security level.

Level I:. Under stable packet loss (Fig. 10(a) and (d)), traditional `P-256` and `x25519` experience significant degradation: mean handshake duration increases to 275-311 ms, coefficients of variation (CV) rise to 2.0-2.8, and outliers rates reach 16-20 %. Despite this degradation, traditional schemes remain the most predictable option with the lowest absolute latency and smallest outlier rate among all tested algorithms.

Hybrid KEMs exhibit higher base handshake times but converge to mean durations of 300-350 ms, with CVs around 2.4 and outlier rates above 18 %, indicating significant overhead and variability. While `mlkem512` in TLS under ideal conditions performs similarly to traditional KEMs, under lossy conditions, it reaches 322 ms mean, CV ≈ 2.4, and 19 % outliers. Notably, in QUIC, `mlkem512` performs markedly better, with a 168 ms mean duration and 15 % outlier rate, outperforming both classical and hybrid KEMs under the same conditions.

In an unstable network (Fig. 11(a) and (d)), TLS and QUIC behavior differ substantially. The mean handshake duration increases significantly in TLS, with traditional KEMS reaching 1300-1500 ms, post-quantum mechanisms around 1880 ms, and hybrid `p256_mlkem512` peaking at 1970 ms. The CV of 3.4–6.3 also indicates substantial spread and heavy tails, despite outliers remaining in a moderate (9-13 %) range. While TLS handshakes degrade considerably under these conditions, QUIC partially mitigates the impact. Indeed, `mlkem512` in QUIC outperforms traditional KEMs both in mean duration and consistency.

Level III:. Under stable packet loss (Figs. 11(b) and 10(e)), the overall behavior closely resembles that of Level I, despite the use of stronger primitives. In TLS, traditional schemes incur mean handshakes durations

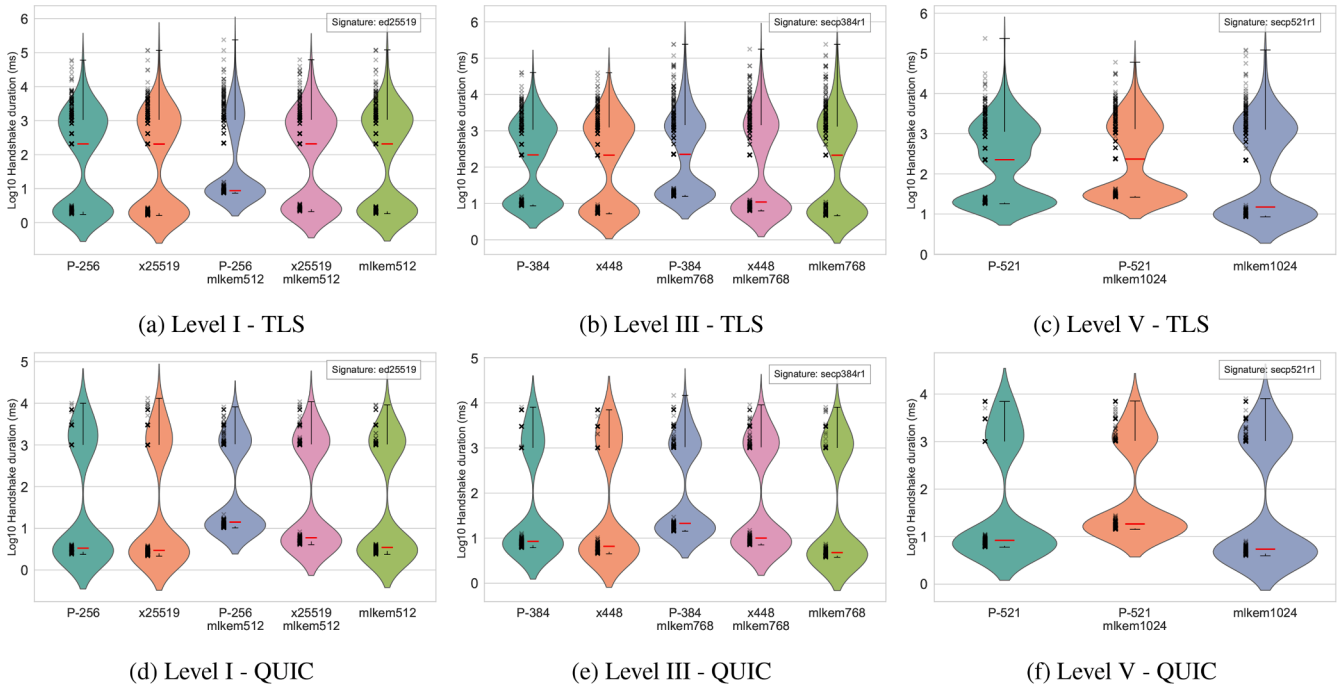


Fig. 11. QUIC and TLS Handshake duration in a unstable network.

of 305-400 ms (vs. 275-311 ms at Level I), with CVs and outlier rates similar to their Level I counterparts. Hybrid and post-quantum KEMs again cluster in the 320-360 ms range, with CVs of 2.3-2.5 and 20-25 % outliers.

At this security level, QUIC exhibits the same loss mitigation effect. Post-quantum, hybrid, and traditional KEMs all shift upward by roughly 50-100 ms compared to Level I. While `mlkem768` remains the fastest under loss, traditional schemes are the slowest (300-435 ms) and hybrid KEMs sit in between. Their CVs (2.2-2.8) and outlier proportions (18-21 %) are similarly stable relative to Level I. Thus, QUIC's advantage over TLS remains intact at this security level under stable loss conditions.

In an unstable scenario (Fig. 11(b) and (e)), all the schemes exhibit increased latency and variability. In TLS, mean handshake time rise by 900-1600 additional milliseconds, with CVs reaching values between 2.3-5.9. In contrast, the impact on QUIC is substantially lower: mean times increase by only 200-300 ms compared to the stable scenario, CVs stay below 2.2, and outlier rates only grow a few percentage points. Consequently, QUIC's resilience advantage is maintained.

Level V:. At this security level, under stable loss conditions, `p521_mlkem1024` is the most efficient KEM for TLS, while `P-521` handshake is the least efficient (see Fig. 10(c)). Under identical conditions, the fastest option for QUIC is `mlkem1024`, followed by `P-521` and finally the hybrid scheme (see Fig. 10(f)). All three handshake times are within a short interval of 220–247 ms.

When network conditions degrade (Fig. 11(c) and (f)), `mlkem1024` in TLS incurs the steepest penalty, with its mean handshake time increasing sixfold. In contrast, `P-521` and `p521_mlkem1024` experience approximately 4.5-fold increases. In contrast, in QUIC, all three schemes degrade by only a factor of 3.5, with `mlkem1024` consistently remaining the fastest option under loss.

These results underscore that TLS amplifies the impact of network instability-particularly for post-quantum primitives-while QUIC's inherent loss resilience mitigates both the absolute latency increase and the performance disparity across classical, hybrid, and post-quantum methods.

6. Conclusion and future work

A seamless and robust transition to the post-quantum web requires a rigorous analysis of the key enabling protocols-TLS and QUIC-and their behavior when combined with novel cryptographic primitives. Our study systematically evaluates their performance with post-quantum and hybrid key exchange mechanisms (KEMs) across three security levels, quantifying trade-offs between latency, bandwidth overhead, and resilience under realistic network conditions. While this work focuses on the empirical performance impact of quantum-resistant primitives, the cryptographic security assurances are derived from the NIST standardization process, which ensures the robustness of the algorithms considered.

While hybrid KEMs serve as a conservative bridge to quantum resistance, our results demonstrate they impose severe performance penalties: 1.5-2.5 times higher latency than traditional schemes in TLS, with amplified cost and variability under adverse network conditions. In contrast, pure post-quantum primitives deliver comparable security guarantees while reducing computational cost by a factor of 1.5-2. Mature schemes like `mlkem` can restore near-classical performance (especially in QUIC) while future-proofing communications. Notably, QUIC exhibits lower sensitivity to stronger KEM algorithms and adverse network conditions, an advantage that becomes increasingly pronounced at higher security levels, reducing computational costs by approximately three-fold. Thus, the transition roadmap should prioritize post-quantum deployment, phasing out hybrids where QUIC mitigates residual post-quantum overhead, and reserving TLS hybrids only for legacy systems.

For future work, we plan to extend our evaluation with hybrid and post-quantum signature algorithms, achieving fully quantum-resistant web stacks. Additionally, we will investigate how these handshake-level performance characteristics translate into applications relying on embedded handshakes, such as VPNs, by measuring their impact on end-to-end throughput and user experience in real-world scenarios. A critical next step is to validate our findings in real-world scenarios. Finally, we intend to reassess our conclusions against forthcoming NIST-approved quantum-resistant standards, ensuring our insights remain relevant as the cryptographic landscape evolves.

CRediT authorship contribution statement

José A. Montenegro: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Ruben Rios:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Javier Bonilla:** Writing – original draft, Visualization, Validation, Software, Resources, Conceptualization.

Data availability

The data is available on https://github.com/montenegro-montes/TLS-QUIC/.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been partially supported by grant PID2022-139268OB-I00 (SecAI project), funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU. Funding for open access charge: Universidad de Málaga / CBUA. The authors are also grateful to Jesús Rodríguez for his technical support.

References

[1] D. Kolesnikov, Next level customer experience with HTTP/3 traffic engineering, 2024, Accessed: 2025-05, https://engineering.zalando.com/posts/2024/06/next-level-customer-experience-with-http3-traffic-engineering.html.

[2] I. Cloudflare, Adoption & usage worldwide, 2025, Accessed: 2025-05-22, https://radar.cloudflare.com/adoption-and-usage.

[3] W3Techs, Usage statistics of default protocol HTTPS for websites, 2025, Accessed: 2025-05, https://w3techs.com/technologies/details/ce-httpsdefault.

[4] Y. Sheffer, R. Holz, P. Saint-Andre, Summarizing known attacks on transport layer security (TLS) and datagram TLS (DTLS), 2015, (RFC 7457). https://doi.org/10.17487/RFC7457

[5] N. Koblitz, A. Menezes, S. Vanstone, The state of elliptic curve cryptography, Des. Codes Cryptogr. 19 (2) (2000) 173–193. https://doi.org/10.1023/A:1008354106356

[6] D. Moody, R. Perlner, A. Regenscheid, A. Robinson, D. Cooper, Transition to Post-Quantum Cryptography Standards, NIST Internal Report (IR) 8547, National Institute of Standards and Technology, 2024. Initial Public Draft. https://doi.org/10.6028/NIST.IR.8547.ipd

[7] C. Gidney, How to factor 2048 bit RSA integers with less than a million noisy qubits, 2025, arXiv: 2505.15917.

[8] D. Stebila, S. Fluhrer, S. Gueron, Hybrid Key Exchange in TLS 1.3, Internet-Draft draft-ietf-tls-hybrid-design-12, Internet Engineering Task Force, 2025. Work in Progress, https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/12/.

[9] N. Standards, NIST Special Publication 1800-16: Securing Web Transactions – TLS Server Certificate Management, Technical Report, U.S. Department of Commerce, Gaithersburg, MD, 2021. https://doi.org/10.6028/NIST.SP.1800-16

[10] P.W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in: Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134. https://doi.org/10.1109/SFCS.1994.365700

[11] C. Gidney, M. Ekerå, How to factor 2048-bit RSA integers in 8 hours using 20 million noisy qubits, Quantum 5 (2021) 433. https://doi.org/10.22331/q-2021-04-15-433

[12] D.J. Bernstein, T. Lange, Post-quantum cryptography—dealing with the fallout of physics success, 2017, (Cryptology ePrint Archive, Paper 2017/314). https://eprint.iacr.org/2017/314.

[13] NIST CSRC, Post-quantum cryptography, 2025, (https://csrc.nist.gov/projects/post-quantum-cryptography). Accessed: 2025-05.

[14] J. Mascelli, M. Rodden, Harvest Now, Decrypt Later: Examining Post-Quantum Cryptography and the Data Privacy Risks for Distributed Ledger Networks, Technical Report, Federal Reserve Board / Finance and Economics Discussion Series, 2025.

[15] P. Ravi, A. Chattopadhyay, J.P. D'Anvers, A. Baksi, Side-channel and fault-injection attacks over lattice-based post-quantum schemes (kyber, dilithium): survey and new results, ACM Trans. Embed. Comput. Syst. 23 (2) (2024). https://doi.org/10.1145/3603170

[16] R. Döring, M. Geitz, Post-quantum cryptography in use: empirical analysis of the TLS handshake performance, in: NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, 2022, pp. 1–5. https://doi.org/10.1109/NOMS54207.2022.9789913

[17] R. Rios, J.A. Montenegro, A. Muñoz, D. Ferraris, Towards the quantum-safe web: benchmarking post-quantum TLS, IEEE Netw. (2025) 1. https://doi.org/10.1109/MNET.2025.3531116

[18] M. Kempf, N. Gauder, B. Jaeger, J. Zirngibl, G. Carle, A quantum of QUIC: dissecting cryptography with post-quantum insights, in: 2024 IFIP Networking Conference (IFIP Networking), 2024, pp. 195–203. https://doi.org/10.23919/IFIPNetworking62109.2024.10619916

[19] M. Raavi, S. Wuthier, P. Chandramouli, X. Zhou, S.-Y. Chang, QUIC protocol with post-quantum authentication, in: W. Susilo, X. Chen, F. Guo, Y. Zhang, R. Intan (Eds.), Information Security, Springer International Publishing, Cham, 2022, pp. 84–91.

[20] M. Raavi, S. Wuthier, X. Zhou, S.-Y. Chang, Post-quantum QUIC protocol in cloud networking, in: 2023 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), 2023, pp. 573–578. https://doi.org/10.1109/EuCNC/6GSummit58263.2023.10188358

[21] D. Stebila, S. Fluhrer, S. Gueron, Hybrid Key Exchange in TLS 1.3, Internet-Draft draft-ietf-tls-hybrid-design-12, Internet Engineering Task Force, 2025. Work in Progress, https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/.