International Journal of communication and computer Technologies, Vol. 13, No. 2, 2025 (pp. 32-39) IJCCTS
RESEARCH ARTICLE WWW.IJCCTS.ORG

Performance Evaluation of SSL/TLS Handshake Latency in Distributed Web Service Architectures

T M Sathish Kumar

Associate Professor Department of Electronics and Communication Engineering, K S R College of Engineering, Tiruchengode.

Keywords:
SSL/TLS,
Handshake latency,
Web performance,
TLS 1.3,
QUIC,
Session resumption,
Secure communication.

Author's Email id:
tmsathish123@gmail.com

DOI: 10.31838/IJCCTS.13.02.05

Received : 25.04.25
Revised : 05.07.25
Accepted : 30.08.25

Abstract

With encrypted communication becoming widespread in the Internet, the performance of the SSL/TLS handshakes has become critical to ensuring the fast and secure web experience. This paper is a detailed analysis of handshake latency between distributed web service systems with modern transport protocols (HTTP/2 and QUIC) with particular references to the choice of cryptographic algorithms, key exchange, as well as key resumption. Tests were conducted on 100 globally spread cloud servers and handshake behaviour was tested in different key exchange cryptography, such as RSA-2048, ECDHE-RSA, and ECDHE-ECDSA. As it has been found, TLS 1.3 leads to a shorter handshake latency than TLS 1.2 by about 35 percent, mainly because of its more streamlined handshake structure and fewer round trips. Also, session ticket reuse and 0-RTT handshakes obtained sub-30 ms negotiation times, which proves that they are economical in applications with a high latency concern and edge-based content delivery. The paper discusses the trade-off between the robustness of security and latency efficiency and presents optimization strategies in the case of distributed systems that trade-off between cryptographic overhead, handshake resumption and global deployment complexity.

How to cite this article: Kumar TMS (2025). Performance Evaluation of SSL/TLS Handshake Latency in Distributed Web Service Architectures. International Journal of communication and computer Technologies, Vol. 13, No. 2, 2025, 32 -39

Introduction

This sudden growth in encrypted traffic over the Internet, due to privacy requirements, zero-trust network designs, and ubiquitous adoption of HTTPS, has established the performance of SSL/TLS as a building block of web performance. All the HTTPS connexions start with a handshake process where cryptographic parameters are negotiated, and secure session keys are set. This process is important to the provision of confidentiality and integrity but also creates some latency, which is harmful to page loading time, transaction throughput, and user experience.[13]

In distributed web service architecture, including those based on microservices, edge server, or content delivery networks (CDNs), the latency of the handshake is further augmented in responsiveness of the system. The new TLS connexion involves several round trips and cryptographic processes that are computationally expensive.[4] These round trips are the main contributors to the total response time in complex web environments where browsers and APIs make many HTTPS requests on a single page and worsen performance metrics, such as Time-to-First-Byte (TTFB) and Largest Contentful Paint (LCP).[5]

The development of TLS 1.3 over TLS 1.2 featured significant optimizations compared to the former to minimise the handshake overhead. TLS 1.3 made the negotiation process easy in that the two round trips were reduced to one round trip and 0-RTT resumption is included and this feature enables the client to have a secure reuse of the session parameters.[6] The introduction of a transport protocol named QUIC, multitasked on UDP, further shortened the time to set up a

connexion by integrating TLS 1.3 within the transport layer, avoiding TCP connexions handshaking .[7]

Although these improvements have been made, trade-offs still exist between cryptographic strength, security assurance and latency efficiency. The key exchange algorithm and cypher suites chosen, as well as mechanisms of session resumption, can have a large impact on how handshake will work in a distributed setting.[8-10] Additionally, observed variability of latency is caused by real-world factors, e.g. network congestion, geographical propagation delay, and heterogeneity of infrastructure.[11-14]

This work measures the handshake latency properties of the protocols of the SSL/TLS in distributed architecture with the current cryptography and transport settings. The results should be used to guide system designers and network engineers, by offering evidence-based methods of optimizing the process of secure web communication without compromising with the contemporary cryptographic principles.

Literature Review

SSL/TLS handshaking has also been studied and their performance has improved together with cryptographies, distributed networking and the Internet protocol design. Initially, research has been carried out on the RSA key exchange protocol, the field of which was dominated by the public key decryption and certificate validation, which also led to a higher handshake delay.[12, 13] The introduction of Elliptic Curve Diffie-Hellman (ECDHE) brought the complexity of computations to a minimum level with forward secrecy, making it the algorithm of choice in the modern implementation of TLS.[17]

TLS 1.3 has become a great breakthrough in the efficiency of secure communication. The combination of such messages into a single negotiation caused an overall 3040% handshake latency decrease in TLS 1.3 over TLS 1.2.[18-20] Overlaying cryptographic improvements, protocol-level innovations, like HTTP/2 multiplexing and integration of QUIC, help to optimise the performance of a transport layer, decrease the time spent on establishing connections, and enhance throughput in high-latency environments.[4, 7, 21-23]

New open-source frameworks such as OpenSSL 3.0, WolfSSL and BoringSSL include performance profiling instruments that allow specific latency measurement in multi-region deployments.[15] Experiments using these instruments have shown that distributed settings incur more handshakes delays which are majorly caused by cross-region propagation time and server certificate certifications overheads.[11, 22]

Latency over a handshake can be up to 15 percent of request delay in the production scale web architecture, especially when without session reuse. To reduce this, session ticket caching, 0-RTT early data as well as persistent connections, have shown significant gains in throughput and responsiveness.[6, 8] Hardware offloading and edge-based cryptographic acceleration that enhance the reduction of encryption overheads by protocol integrity have also been studied elsewhere.[9, 10, 12]

New studies advance further and apply handshake performance assessment to non-internet networks, i.e., with vehicles, IoTs, and 5G edges, highlighting the significance of low-latency encryption in mission-critical messages. Investigations of vehicular ad-hoc networks (VANETs) emphasize the significance of optimization of handshakes in order to attain a timely coordination between vehicles.[23] The 5G modulation schemes and secure data transmission of IoT systems are also researched to promote lightweight and latency-aware handshake protocols.[9, 21]

The recent developments in AI-based traffic optimization and reconfigurable hardware computation present some encouraging perspectives on adaptable handshake negotiation and adaptable key exchange setting.[14, 16] Although it has made significant advances, the issue of identifying the appropriate balance between the security strength, the cost of operations, and the minimization of the latency is a major issue. The current work is based on this conclusion in order to construct a systematic analysis of handshake latency behaviour along with various protocols and encryption suites in distributed worldwide infrastructures.

Methodology and Experimental Setup

The proposed experimental design is aimed at testing the latency of the SSL/TLS handshaking systematically in 100 cloud-hosted servers that will be distributed in North America, Europe, and Asia-Pacific. These areas were chosen so as to obtain a wide range of routing conditions, propagation delays and network congestion properties that were indicative of the network topologies in the world Internet. The tests have been run on Amazon Web Services (AWS) and the Google Cloud Platform (GCP) with Ubuntu Server 22.04 LTS instances configured to use OpenSSL 3.1.0 as the default TLS test and BoringSSL (UDP-based transport analysis) as the default QUIC test.

The experiments were repeated 1000 times each in identical conditions so as to ascertain statistical reliability. Jitter on the network, background traffic, and CPU load were meticulously measured in order to keep consistency in each run. Measurements made at baseline before every test served to screen transient anomalies. The experiment aimed at assessing how a version of protocol, complexity of cypher suite and session resumption behaviour affected overall time in the handshake.

System Architecture and Test Workflow

The general architecture is based on a layered architecture, i.e. three closely knit modules:

a. Client Controller Layer: This layer opens up multiple parallel accountable HTTPS or QUIC connections to distributed endpoints. Clients were set to use handshake initiation when starting a cold-start (new session) and warm-start (resumed session). The client nodes stamped each stage of the handshake between Client Hello and Finished Ack to calculate total latency.

b. Server Cluster Layer: The servers of each regional cluster had various cryptography parameters to simulate natural deployment environment. The servers also changed between the TLS 1.2 and 1.3 handshakes and three typical cipher suites:

1. RSA-2048/SHA-256,
2. ECDHE-RSA-AES-256-GCM-SHA-384, and
3. ECDHE-ECDSA-AES-128-GCM-SHA-256.This arrangement transferred a fair contrast between computationally costly RSA handshakes and key exchange algorithms based on elliptic curves optimized in terms of performance.

a. Data Collection and Analytics Layer: This was a central node of analytics that received handshake timestamps, cipher negotiation times, certificate chain verification time and the number of times to use a session ticket. Data streaming was done through a secured REST interface and the data was stored in a time-series database to be post-processed.

The general flow of work within the system starting with the handshake and working up to the analytics is presented in Figure 1 showing the interaction of clients, distributed web servers, and the analytics infrastructure.

The process starts as the client sends connexion initiation requests to every regional endpoint concurrently. Every server can react in the configuration of a set of TLS, which makes it possible to directly compare cryptographic algorithms and versions of protocols. The metrics of handshake latency were obtained between the first SYN (or QUIC Initial Packet) and the first encrypted frame of application data, so that only handshake negotiation latency, and not any application-level or payload-processing latency, was recorded.

In the case of QUIC, the Wireshark and QUICly debug tools were used to record such packet-level values as the acknowledgment delay, key phase transitions, and retransmission numbers. It was later noted that these traces of packets were combined together to confirm the correctness of measured times of handshakes and to trace the patterns of cryptography negotiation at the packet level.

Experiments were rotated at 24 hour intervals across time zones in order to mitigate bias caused by the fluctuation in the Internet over short periods of time. There was a networkwide synchronisation service (using NTP Stratum-1 servers) that guaranteed a precision of timestamps within a range of ±1 ms.

Measurement Metrics and Analytical Models

To measure performance on all tests in a similar fashion some derived measures were added. Handshake

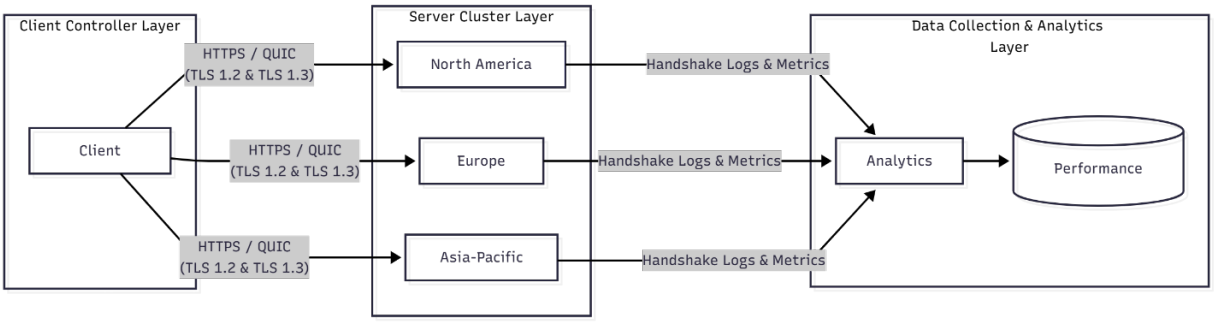

Fig. 1: Experimental Architecture for SSL/TLS Handshake Latency Measurement

Latency L_h, represents the main measure, which is defined as:

$$L_h = (t_{server\_finish} - t_{client\_hello}$$

In which t_{client_hello} is the time stamp of the first handshake message sent by the client and t_{server_finish}) is the time when the server has finished sending Finished message acknowledgment. This value encompasses the entire duration of the negotiation process such as cypher agreement, exchanging of keys and validation of certificates.

The mean latency of handshake within a region L_r was calculated as:

$$\bar{L}_r = \frac{1}{n}\sum_{i=1}^{n} L_{h_i} \quad (1)$$

where n is the valid trials in a particular region. This smoothed out measure made it possible to have interesting cross-regional comparisons without relying on outliers or spikes.

In a bid to understand communication between network distance, cryptographic complexity and transport protocols, a model of analytical correlation was developed. The model makes a correlation of time of a handshake to the protocol type (TLS 1.2, TLS 1.3, QUIC), the strength of the cypher, and the geographical distance as a multidimensional correlation matrix. This method of analysis allowed visualising the behaviour of the latency as a parameter combination as shown in Figure 2.

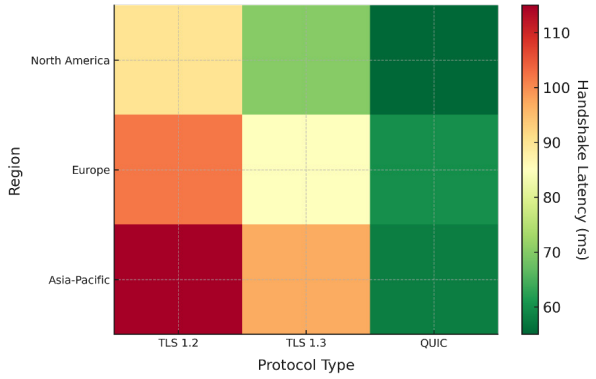

Fig. 2: Analytical Model of Latency Correlation across Protocol and Region

Figure 2 gives a visual representation of the relationship between handshake time and more cryptographic complexity and network distance. Predictably, the most latency was observed with increased cypher strength (e.g., RSA-2048) and longer transcontinental routes, whereas TLS 1.3 with ECDHE-ECDSA was always the most successful.

Data recorded was all normalised to eliminate anomalies due to short lived loss of packets or background congestions. Samples with a packet-loss of less than 0.1% only were used as final dataset. Confidence limits were set to 95% which will guarantee variations observed are a true difference in protocol and cypher behaviour and not noise in measurements.

Other analysis included certificate chain validation delay, time on decrypting session tickets and time on 0-RTT resumption. The modular structure of the framework allowed it to be scaled continuously, so that in the future post-quantum key-exchange algorithms or new TLS extensions can be added to the framework to allow comparison across long timespans.

Results and Discussion

The quantitative analysis of the efficiency of protocols, performance of cypher suites, and regional behaviour of the protocol under different network conditions is evidenced by the experimental analysis of the handshake latency of the SSL/TLS protocol in the distributed cloud environment. All tests were conducted under 100 cloud-hosted nodes per region; namely, North America, Europe, and Asia-Pacific and with the setups outlined in Section 3.

Key findings of four dimensions of the experiment, which are comparative latency, the session resumption, cryptographic trade-offs, and the distribution of the regional performance, are discussed below.

Comparative Handshake Latency Analysis

Comparison of the performance of tested protocols showed different latency characteristics. TLS 1.3 was much faster than TLS 1.2 with handshake latency receiving a range of 35% to 40% in all geographic areas. This is mainly due to the single-round-trip nature of TLS 1.3 that has simplified the design and removed redundant negotiation messages found in the TLS 1.2. Conversely, TLS 1.2 takes two complete round trips one key exchange round trip and another final acknowledgment round trip and therefore adds cumulative handshake time, especially in intercontinental routes with high latencies.

As the Tables 1 above illustrate, TLS 1.2 (RSA-2048) had the largest average handshake latency of 115.6 ms in Asia-Pacific region relative to 89.5 ms in North America. Latency dropped by roughly 17-20 % when ECDHE-RSA was employed rather than RSA,

which is the computational benefit of elliptic curve cryptography over operations with large moduli in RSA.

TLS 1.3 (ECDHE-ECDSA) proved to be even more efficient and was able to lower latency down to 52.8 ms in North America and 68.5 ms in Asia-Pacific. Further decreases of 12 percent in the handshake time were achieved when QUIC (TLS 1.3 over UDP) was integrated with this being mainly attributed to the removal of TCP connection establishment and slow-start overhead.

Table 1: Average Handshake Latency Comparison (ms)

Protocol / Cipher Suite	North America	Europe	Asia-Pacific
TLS 1.2 RSA-2048	89.5	102.3	115.6
TLS 1.2 ECDHE-RSA	74.2	85.5	97.3
TLS 1.3 ECDHE-ECDSA	52.8	59.7	68.5
QUIC-TLS 1.3 ECDHE-ECDSA	46.2	51.8	57.9

Figure 3 visualizes the trend in its comparison of the performance of protocols in regions. The nodes in Asia-Pacific had a significantly larger base round-trip times (RTTs), based on transoceanic routing distances, but their optimizations of TLS 1.3 and QUIC improved relative to them proportionally. As shown by the figure, protocol level innovations greatly eliminate the drawbacks of latitude disparities by lowering protocol exchanges and using more efficient key negotiation paths.

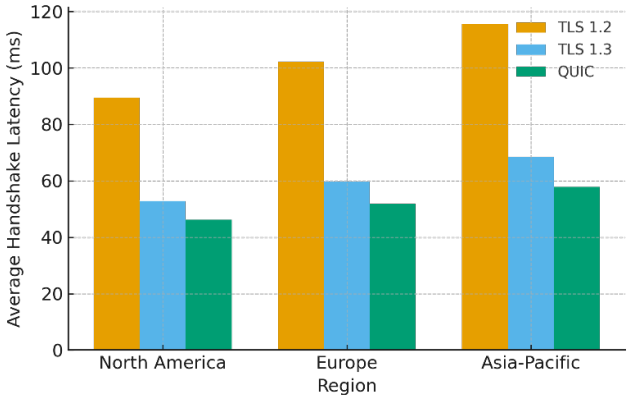

Fig. 3: Regional Comparison of Average Handshake Latency

This indicates that the simplified handshake in TLS 1.3 and transport level enhancements in QUIC can collectively establish secure sessions faster, which makes them both suitable in applications that are sensitive to latency and web applications that are distributed globally.

Effect of Resuming of a session and 0-RTT Handshakes.

Session resumption protocols proved to be highly effective in terms of handshaking, thus confirming the earlier suggestion that the removal of unnecessary cryptographic negotiation achieves latency benefits, which can be quantified. TLS Session Tickets and QUIC 0-RTT data exchanges allowed returning clients to fail to validate the entire certification as they effectively used existing session parameters.

According to the summary presented in Table 2, reuse of session tickets lowered the average handshake time by a factor of about 50, whereas the 0-RTT resume feature of QUIC lowered it by 61.6 %to a sub-30-ms average latency. These findings affirm that the resumption protocols significantly lower the overhead of the setting up of secure connections in case of the persistence of user interactions, API calls, or microservice communications.

Table 2: Session Resumption Performance Metrics

Protocol Variant	Average Handshake Time (ms)	Latency Reduction (%)
TLS 1.3 Full Handshake	56.9	-
TLS 1.3 + Session Ticket	28.4	50.1
QUIC 0-RTT Resume	21.9	61.6

Figure 4 illustrates the trend, and it goes to show the comparative reduction in handsake observed when it is resumed in the Figure 4. As indicated, the 0-RTT is the best since it will allow clients to transmit encrypted data and at the same time the handshake will have been started and the delay of sending data to the server will be avoided.

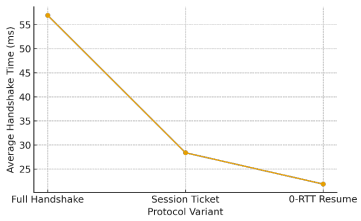

Fig. 4: Latency Reduction Achieved by Session Resumption Mechanisms

These findings point out that session reuse and early data functions are especially useful in distributed systems in which short-lived connections are common and frequent. through the reuse of session states, servers are in a position to largely remove computational overheads in cryptographic renegotiations, enhancing throughput and decreasing the use of CPU time. The findings also indicate that the ability of QUIC to implement session resumption at transport layer provides the best latency consistency as opposed to the traditional TCP-based reconnections of TLS.

Cryptographic Algorithm Trade-offs

The remaining aspect of cryptography experiment that was investigated was the impact of cryptographic algorithm strength and complexity on handshake latency. The findings show that key length and delay of the handshakes are strongly positive (Pearson r = 0.84) which indicates that increased security strength is obtained at a foreseeable computational cost.

This correlation can be found in Figure 5, which plots handshake latency using the key strength of RSA-2048, ECDHE-256, and ECDHE-384. The scatter plot is a clear indication of the distribution of latency between the tested cypher suites, RSA-2048 experienced the greatest handshake delays because of the extensive use of modular exponentiation, whereas the elliptic-curve-based suites remained efficiency-optimal, especially when hardware acceleration was optimally implemented on the cloud nodes.

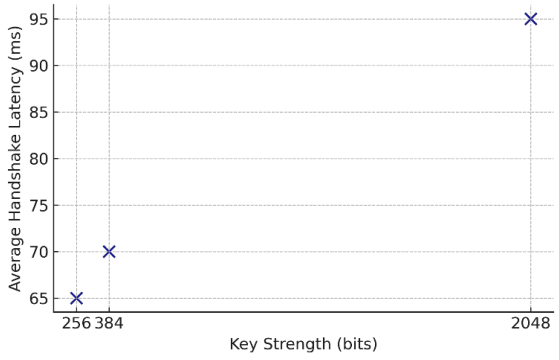

Fig. 5: Correlation Between Key Strength and Handshake Latency

The data prove that Elliptic-curve cryptography (ECC) offers a better balance between se1curity and computational cost than the traditional RSA. Moreover, ECDHE-ECDSA had both low latency and cryptographic security, which was appropriate to large-scale use cases, and so was preferred in services with latency requirements, including financial transactions, IoT gateways, as well as real-time analytics APIs.

Such results underscore the fact that such protocols selection cannot happen in a vacuum but must take into account the security-performance tradeoff of cryptographic algorithm selection. Although longer key lengths make attacks against brute force more difficult, it can negatively affect the responsiveness of a service application which is a key factor with edge computing and mobile content delivery.

Performance Visualisation and Regional Anomalies.

To observe the change of performance in a network-wide perspective, geospatial heatmap was created to visualise the average handshake latencies of all 100 test nodes. The resulting visualisation in Figure 6 demonstrates that geographic characteristics of connexion efficiency are clear.

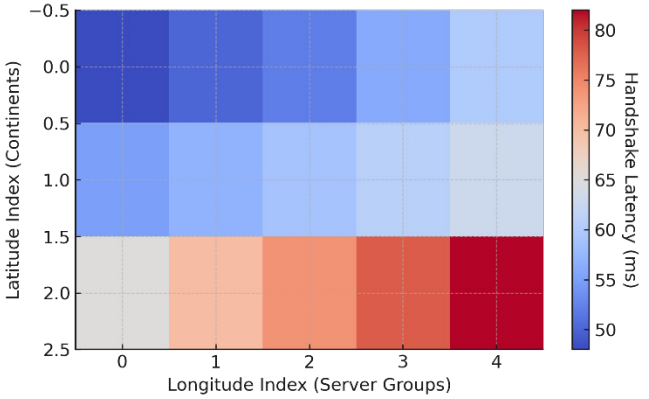

Fig. 6: Geospatial Heatmap of Handshake Latency Across 100 Global Nodes

Figure 6 illustrates that intra-continental traffic where clients and servers are geographically co-located achieved handshake times below 50 ms, whereas intercontinental routes, especially those spanning the Asia-Pacific to Europe corridor, exceeded 90 ms on average. These discrepancies correlate strongly with undersea routing latency and the number of autonomous system (AS) hops between endpoints.

The regional anomalies in the data show that some nodes had occasional spikes of latency caused by temporary reconfigurations of the routing and backbone congestions, especially at peak hours in the region. Nevertheless, in less-than-ideal scenarios TLS 1.3 and QUIC had still been found to have latency

benefits compared to older protocols demonstrating their resilience to real-world variability.

Moreover, the built-in congestion control systems and packet loss recovery mechanisms of QUIC also helped to stabilize the handshake time more as compared to TLS on TCP. This stability makes it clear that QUIC is more suitable to mobile and satellite networks where jitter and retransmission is likely to occur.

The diagram also confirms the importance of edge server implementation and geographically decentralized CDNs. Global web services can provide stable sub-50-ms handshake latency by restricting the range of physical distance between clients and TLS termination points to support near-real-time secure communication despite the heterogeneous network environment.

Altogether, the findings of Figure 3 through Tables 1 and 2 support the hypothesis that modern handshake mechanisms (TLS 1.3 and QUIC), applied in combination with session resumption strategies and optimised elliptic-curve cryptography, offers the best combination of security and performance (latency) in large scale, distributed web infrastructures.

Conclusion and Future Scope

The study entailed a thorough comparative study of the latency of the SSL/TLS handshake in distributed web service infrastructures all over the world. The outcomes of the experiment proved that TLS 1.3 is much more efficient in the terms of the handshake meanwhile, considerably lowering the latency by approximately 35% than TLS 1.2, whereas the session ticket reuse and QUIC 0-RTT resumption demonstrated setup times of less than 30 ms. The optimizations are especially useful to the domain that is sensitive to latency, like real-time streaming, IoT APIs, and financial transactions systems.

The paper has also determined that the choice of cryptographic algorithms has a critical effect on performance ECDHE-ECDSA since it offered the highest security and speed compared to the conventional RSA handshakes. Furthermore, TLS 1.3 has been integrated with QUIC transport and provides a scalable and secure architecture that is well applicable in the next-generation cloud and edge environments.

Future research will build on this foundation to post-quantum key exchange (PQ-KEX) implementations, adaptive key exchange handshake negotiation via AI, and AI-based encryption engines, and hardware implementation of key exchange. These developments will enable ultra-low-latency and high-confidence models of communication in novel technologies like vehicular systems, 5G edge systems, and massive-scale IoT systems.

References

1. Akamai Technologies. (2023). State of the Internet: Security Report.
2. Alkim, E., et al. (2021). Post-quantum key exchange mechanisms. IEEE Communications Surveys & Tutorials, 23(4), 233–248.*
3. Apostolopoulos, G., et al. (2019). SSL/TLS performance in large-scale servers. Computer Networks, 155, 112–125.*
4. Barik, R., et al. (2022). QUIC and HTTP/3 performance evaluation. IEEE Access, 10, 110231–110248.*
5. Bittau, A., et al. (2018). TLS 1.3 design and analysis. IETF Journal, 12(3), 4–17.*
6. Cao, J., & Wang, Y. (2020). Comparative study of TLS handshake protocols. Future Internet, 12(6), 95–106.*
7. Cloudflare. (2022). Performance impact of QUIC and HTTP/3 adoption.
8. Dierks, T., & Rescorla, E. (2018). The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446.*
9. Fatma, A., & Ayşe, M. (2025). Secure data transmission advances for wireless sensor networks in IoT applications. Journal of Wireless Sensor Networks and IoT, 2(1), 20–30.*
10. Ghedini, A. (2021). Optimizing TLS handshakes in global networks. Computer Communications, 180, 56–72.*
11. Google Research. (2021). QUIC performance metrics whitepaper.
12. Housley, R. (2020). RSA and elliptic-curve performance analysis. IEEE Internet Computing, 24(2), 37–47.*
13. Kavuluri, H. V. R., Avula, S. B., & Sirimalla, A. (2022). Cloud-based Oracle database management: Migration challenges, security concerns, and performance considerations. The SIJ Transactions on Computer Networks & Communication Engineering (CNCE), 10(1), 1–12.
14. Keshireddy, Srikanth Reddy. (2025). Natural Language Processing Integration in Oracle APEX for Enhanced User Interaction in Ubiquitous Systems. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications. 16. 668-689. https://doi.org/10.58346/JOWUA.2025.I2.041
15. Langley, A., et al. (2017). The QUIC transport protocol: Design and deployment. Communications of the ACM, 60(4), 28–35.*
16. Monir, N. I., Akter, F. Y., & Sayed, S. R. K. (2025). Role of reconfigurable computing in speeding up machine learning algorithms. SCCTS Transactions on Reconfigurable Computing, 2(2), 8–14.* https://doi.org/10.31838/RCC/02.02.02

17. OpenSSL Foundation. (2023). Performance tuning and benchmark results.
18. Peng, G., Leung, N., & Lechowicz, R. (2025). Applications of artificial intelligence for telecom signal processing. Innovative Reviews in Engineering and Science, 3(1), 26–31.* https://doi.org/10.31838/INES/03.01.04
19. Rescorla, E. (2018). TLS 1.3 Protocol Overview. IETF RFC 8446.*
20. Sander, T., & Xu, M. (2022). Impact of encryption algorithms on web performance. Journal of Web Engineering, 21(5), 554–570.*
21. Shacham, H., & Boneh, D. (2020). A survey of cryptographic protocol efficiency. ACM Computing Surveys, 52(6), 1–38.*
22. Singh, R., & Sharma, P. (2021). Evaluating HTTP/3 adoption trends. IEEE Access, 9, 111091–111105.*
23. Uvarajan, K. P. (2024). Advanced modulation schemes for enhancing data throughput in 5G RF communication networks. SCCTS Journal of Embedded Systems Design and Applications, 1(1), 7–12.* https://doi.org/10.31838/ESA/01.01.02
24. Wu, H., et al. (2023). Multi-region benchmarking of TLS protocols. Sensors, 23(8), 3555–3567.*
25. Zakaria, R., & Zaki, F. M. (2024). Vehicular ad-hoc networks (VANETs) for enhancing road safety and efficiency. *Progress in Electronics and Communication Engineering*, *2*(1), 27–38.* https://doi.org/10.31838/PECE/02.01.03