Lightweight Identity-Based Re-Authentication for Supporting Post-Quantum Security in 5G

Manish Paudel*, Maryna Veksler, Kemal Akkaya
Advanced Wireless and Security Lab, *Virginia Commonwealth University*, Richmond, VA USA 23220
Email: {paudelm, vekslerm, akkayak}@vcu.edu

***Abstract*—The rapid growth of 5G wireless technology has transformed connectivity, offering exceptional bandwidth, ultra-low latency, and massive IoT device connectivity. However, as quantum computers are progressing, resistance against attacks from such computers becomes a mandatory requirement for all critical infrastructure, necessitating efficient post-quantum cryptography (PQC) implementations for 5G-based IoT devices with limited resources. For instance, User Equipment (UE) re-authentication due to frequent handovers and mobility events is a daily operation that already comes with some overhead, which is not attractive to be used for IoT UE devices. Incorporating heavier PQ solutions into these re-authentications will lead to even more additional communication and computation overhead that may hinder PQ deployment in next generation networks. As such, this paper introduces a novel, lightweight approach for integrating PQC in 5G IoT authentication by proposing a custom identity-based session resumption mechanism without compromising interoperability with existing protocols. This approach prevents replay attacks and ensures perfect forward secrecy (PFS) by utilizing customized identities that are dynamically generated based on device-specific parameters, combined with intelligent server-side caching of quantum-resistant cryptographic materials that eliminate the need for full PQC computations during subsequent re-authentications. We demonstrate through realistic 5G experimentation that the proposed solution significantly lowers authentication overhead while providing quantum-resistant security.**

***Index Terms*—Post Quantum Cryptography (PQC), 5G, IoT, TLS, Session Resumption, Perfect Forward Secrecy (PFS)**

I. Introduction

5G mobile technology enables an integrated network that is designed to connect virtually everyone and everything through its key features: Enhanced Mobile Broadband (eMBB), Ultra-Reliable and Low-Latency Communication (URLCC), and massive machine-type communications (mMTC). This improved connectivity ensures the efficient functioning of billions of IoT devices within the Internet of Everything (IoE) framework. With an estimated 40 billion connected devices by the end of 2030 [1], the integration of 5G and IoT has the potential to drive a technological revolution, transforming smarter cities, industries, and digital ecosystems.

Workshop on Security and Privacy of Next-Generation Networks (FutureG) 2026
23 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-9-7
https://dx.doi.org/10.14722/futureg.2026.23079
www.ndss-symposium.org

The current 5G security framework primarily relies on the 5G Authentication and Key Agreement (5G-AKA) protocol and Extensible Authentication Protocol (EAP-AKA') [2]. However, these protocols not only exhibit inherent vulnerabilities but are also expected to be adversely affected by advancements in quantum computing [3], [4], [5], [6]. Therefore, the 3GPP mandates TLS to secure the Service Based Architecture (SBA) and supports EAP-TLS as an alternative to 5G-AKA for IoTs, allowing for certificate-based security where SIM management is impractical. Nonetheless, TLS is also prone to threats from quantum computers [7], [8]. In response to these threats, NIST's PQC standardization efforts released standards for quantum-resistant algorithms, including CRYSTALS-Kyber (ML-KEM) for key encapsulation, CRYSTALS-Dilithium, Falcon, and SPHINCS+-SHA256 for digital signatures. [9]. This led to exploration of PQ-TLS solutions for 5G with different objectives [10], [11].

Most of these research efforts, however, have focused primarily on reducing the initial TLS handshake overhead [12], [13], overlooking the critical challenges of frequent re-authentication caused by high mobility and dynamic configurations from the UE side in 5G. Specifically, *unlike traditional networks, 5G IoT devices must perform frequent re-authentication throughout their operational lifecycle*. These authentication events occur when the UE transitions between network slices or performs periodic registration updates. Such updates typically happen within a timer-based interval ranging from every 54 minutes to 12 hours, depending on operator settings [14]. Additionally, authentication may be triggered when the UE roams between base stations or experiences temporary disruptions, such as moving through tunnels [15]. Each authentication process requires the execution of another 5G registration request. Given the sheer number of 5G IoT devices (e.g., drones, vehicles, robots, etc.), this requirement can lead to significant overhead for both the devices and the network (core or gNB). Specifically, repeated authentication attempts contribute to increased signaling overhead, latency, and computational/energy overhead for already resource-constrained 5G IoT devices, draining their batteries. *These challenges are further exacerbated when PQ solutions are to be considered for the re-authentication procedure*. Due to overhead that comes from large key/certificate sizes or computational overheads, PQ solutions for re-authentication will make the overhead situation even worse. As a result, both robust and lightweight solutions are needed for large-scale integration of

resource-constrained IoT devices in 5G that are also resistant to quantum attacks.

One might consider that utilizing TLS 1.3 session resumption with session tickets or pre-shared keys (PSK) could help clients authenticate without repeatedly exchanging large certificates [16]. Unfortunately, standard TLS 1.3 session resumption presents security risks that are particularly problematic in the context of frequent 5G IoT re-authentication. First, *the protocol is vulnerable to replay attacks*, where an attacker can capture and resend valid session tickets, potentially causing duplicate operations or unauthorized access [17]. Second, *standard TLS 1.3 session resumption lacks Perfect Forward Secrecy (PFS)* [18] as the Session Ticket Encryption Key (STEK) is not rotated per session. Third, *there is no binding of session tickets to client identity*, allowing an attacker with a valid session ticket to impersonate the legitimate client. Additionally, any new solution must be *compatible with existing TLS standards*, as changes would necessitate impractical updates across various infrastructures. This need for backward compatibility further complicates the integration of PQC solutions, which need to coexist with the current TLS mechanism. Therefore, new approaches are needed to ensure PQ-TLS is viable for 5G IoT devices throughout their deployment lifecycle while addressing these fundamental security gaps in session resumption.

We propose a lightweight approach for 5G IoT that introduces a custom *identity-based session resumption mechanism* that enhances the standard TLS 1.3 with additional security layers while *maintaining full compatibility with the existing TLS 1.3 standard*. Our key contributions are as follows.

(1) PQ-enabled session resumption for frequent re-authentication: IoT devices perform expensive PQC operations only once during the initial handshake, with subsequent re-authentication events using lightweight session resumption. This makes quantum-resistant security practical for resource-constrained devices that must re-authenticate dozens of times daily due to mobility and network changes.

(2) Replay-resistant architecture: We embed custom client identity, timestamp, and high-entropy random values within TLS ClientHello as extension data, combined with server-side replay cache validation. This prevents replay attacks and ensures identity binding during session resumption without protocol modifications.

(3) PFS with quantum resistance: The server rotates session ticket encryption keys (STEK) with each reconnection using quantum-resistant key encapsulation, ensuring that compromise of current keys does not jeopardize past sessions. Each reconnection issues a new session ticket with freshly derived quantum-resistant cryptographic material.

(4) Secured 0-RTT data with identity binding: Our approach supports early data processing only after rigorous validation of custom extension data and replay cache checks. Unlike standard TLS 1.3, we ensure quantum-secured 0-RTT communication with verified client identity.

We validated our solution through experiments in a local 5G testbed built using open-source implementations (Open5GS for 5G core and UERANSIM for UE/gNB) and deployed a similar setup on Google Cloud Platform. Our approach achieves substantial performance improvements: for SPHINCS+-SHA256, session resumption delivers 2.9× faster authentication, 47% bandwidth reduction, 17% fewer signaling messages, and 16% energy savings compared to full PQC handshakes, with similar gains observed for other PQC algorithms (Falcon-512, Dilithium2). Additionally, we have included an extensive security analysis of the proposed method, highlighting its resilience against various potential threats.

Section II covers background (5G architecture, UE registration and re-registration, TLS, and PQC). Section III reviews the existing literature on 5G security. Section IV details our PQ-Enhanced TLS Resumption with Custom Authentication. Section V presents security analysis. Performance evaluation is given in Section VI. Finally, Section VII concludes the paper.

II. Background and Preliminaries

This section briefly describes the fundamental concepts and technologies used in our work.

A. 5G Background

5G, the fifth generation of cellular networks, delivers ultra-high data rates, low latency, and massive connectivity to support applications such as autonomous vehicles, smart cities, and the IoT. It uses an SBA and technologies such as software-defined networking (SDN), Network Function Virtualization (NFV), network slicing, edge computing, and enormous Massive Multiple-Input Multiple-Output (MIMO) for enhancing performance and offering dynamic network management. 5G features a separation of the User Plane and the Control Plane, improving traffic management. Key functions and general architecture of 5G are shown in Figure 1.

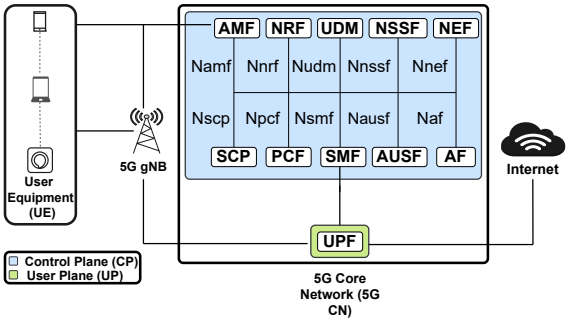

Fig. 1: 5G-Architecture

B. UE-Registration and Re-registration process

UE registration is a core process in 5G networks that enables secure connectivity to 5G core. During registration, the UE transmits a permanent subscription identifier (*SUPI*), which is converted into a Subscription Concealed Identifier (SUCI) using public key encryption in accordance with 3GPP standard. The registration supports initial access, mobility management, periodic updates, and emergency registration.

Re-authentication is essential to maintain secure connectivity in 5G IoT environments, particularly under mobility and network changes. As IoT deployments scale, challenges in

authentication, mobility, and session continuity necessitate re-authentication in various scenarios, including:

I. *Extended Disconnection*: UE is disconnected for a specified duration set by the network operators.

II. *Network slice-specific re-authentication*: AMF interacts with the Network Slice Selection Assistance Function (NSSAF), which communicates with a third-party Authentication Authorization and Accounting (AAA) server.

III. *Mobility*: High-mobility IoT devices (mobile, drones) trigger re-authentication when crossing tracking area boundaries. UAV studies show handovers every 2-5 minutes [19].

IV. *Registration on a different Slice*: UE attempts to register for a second slice while already registered to another.

V. *Periodic Re-authentication*: Network operators periodically update the tracking area for UE within a specified time frame of minutes to hours determined by Non-Access Stratum (NAS) security timers.

VI. *AMF initiated Re-authentication*: AMF triggers Re-authentication due to subscription or operator policy changes (less frequent).

C. Transport Layer Security (TLS) and Session Resumption

The TLS handshake establishes trust between a client and a server through message exchanges that create a shared key for encrypting communications.Figure 2 shows the TLS handshake with session ticket extension. **Session Resumption** [20] allows clients to resume sessions using tickets in the *SessionTicket* extension.

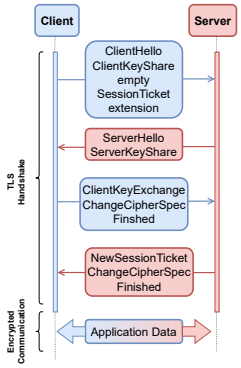

Fig. 2: Traditional Session Resumption

TLS Key Generation and Session Resumption Mechanism: TLS 1.3 uses a hierarchical HMAC-based extract-and-expand key derivation function (HKDF)-based approach to derive cryptographic secrets [16] and pre-shared key (PSK). The master_secret (ms) is formed through early_secret (es) and handshake_secret (hs) as shown in Equations 1-3:

$$\text{early\_secret(es)} = \text{HKDF-Extract}(0, \text{PSK}) \quad (1)$$

$$\text{handshake\_secret(hs)} = \text{HKDF-Extract}(\text{es}, \text{SS}) \quad (2)$$

$$\text{master\_secret(ms)} = \text{HKDF-Extract}(\text{hs}, 0) \quad (3)$$

Traffic secrets and resumption keys (RMS, PSK) are derived using HKDF-Expand-Label as shown in the Equations 4 & 5.

$$\text{rms} = \text{HKDF-Expand-Label}(\text{ms}, \text{"res master"}, \text{h}_{\text{ch...fin}}, \text{l}) \quad (4)$$

$$\text{PSK} = \text{HKDF-Expand-Label}(\text{rms}, \text{"resumption"}, \text{n}_\text{t}, \text{l}) \quad (5)$$

Finally, the session ticket is encrypted using a Server Ticket Encryption Key (STEK), which is periodically rotated. For session resumption, TLS uses a hierarchical derivation ensuring key separation and uses a similar HKDF-Expand-Label to calculate all the traffic secrets and client keys. The major difference between a full handshake and session resumption is that the value of PSK is 0 in the initial full handshake, whereas PSK is determined through RMS for session resumption [16].

D. PQC Algorithms

The rise of quantum computers presents a significant threat to modern public key cryptography [21]. Algorithms such as RSA and Elliptic Curve Cryptography (ECC) rely on the difficulty of mathematical problems like integer factorization and the discrete logarithm problem. However, quantum computers can efficiently solve these problems, compromising current systems, once sufficiently powerful quantum machines become available [7] [22]. To address this imminent threat, PQC aims to develop algorithms resistant to both quantum and classical attacks. The National Institute of Standards and Technology (NIST) is leading efforts to standardize PQC algorithms, having selected two lattice-based signature schemes, CRYSTALS-Dilithium and Falcon, a stateless hash-based scheme, SPHINCS+SHA [23], and a lattice-based key encapsulation mechanism, CRYSTALS-Kyber/ML-KEM.

These signature schemes are essential to TLS for authenticating the server and client during the handshake process, as explained in II-C. By substituting the traditional signature algorithm with PQC alternatives, we can ensure that the TLS handshake remains quantum secure.

III. Related Work

5G Security and Authentication: Current 5G security (5G-AKA, EAP-AKA') has significant weaknesses [2] [3] [4] including linkability attacks [24] and privacy attacks [25]. While 3GPP recommends EAP-TLS [2], many open-source 5G implementations have yet to incorporate EAP-TLS in the data plane. Recent PQ-TLS work [10], [11] focuses on initial handshakes, not frequent re-authentication. Our research takes this a step further by not only integrating PQ-TLS but also considering its customization for re-authentication purposes in the context of IoT devices.

PQC in Resource-Constrained Environment: Quantum threats significantly risk classical cryptographic algorithms protecting IoT infrastructure. PQC solutions require higher computational demands in resource-limited IoT systems [26], increasing energy consumption and runtime by 1.5-7× [27], with significant performance impacts for 5G IoT [10]. We address this by using custom session resumption to minimize PQC overhead, performing intensive operations only during initial handshake and using lightweight resumption for subsequent re-authentications.

Anti-Replay Techniques: TLS session resumption [16] enhances performance but poses security risks like replay attacks [17]. Research by [17] and [28] reveals significant vulnerabilities in session ticket implementations, including weak STEKs, and discusses the advantages and vulnerabilities associated with TLS session resumption.

In addition to conventional methods, researchers have proposed several alternative techniques to address the security-performance tradeoff, such as the Bloom filter-based approach [29]. However, this method is incompatible with the standardized TLS protocol [30] and may result in rejection of the resumption request, forcing a full handshake. The QUIC protocol uses address validation tokens and transport parameters

to manage 0-RTT data with anti-replay protection directly at the transport layer [31], but faces compatibility issues with existing infrastructure. The puncturable pseudorandom function (PPRF) [18] offers PFS and an anti-replay mechanism, but it requires significant computational resources and modifications to TLS. The rTLS [32] extension relies on client-side storage, which is challenging for resource-constrained devices, and its security analysis is limited to bounded sessions.

With no single solution ideal for all scenarios, our approach provides replay protection, PFS, and identity binding through server-side validation, all while maintaining compatibility with standard TLS 1.3 and providing quantum resistance.

IV. Proposed Approach

A. Problem and Motivation

The 5G security architecture necessitates frequent re-authentication for IoT devices due to mobility events, which can strain both the devices and the 5G core network. The emergence of quantum computing complicates this further, requiring the integration of PQC into TLS [7]. Currently, open-source 5G implementations lack TLS support across all planes, making it crucial to efficiently incorporate PQ-TLS to reduce re-authentication overhead for IoT devices.

Instead of restarting authentication for each mobility event, session resumption can help minimize computational and communication overhead. However, traditional TLS session resumption is prone to replay attacks and lacks PFS [18], especially in mobile contexts. While there are alternative methods [18], [29], [31], [32], they often require protocol redesign or introduce practical limitations, making them impractical for large-scale 5G IoT deployments.

In our work, we address this significant security gap by proposing a novel replay-resistant PQ-TLS session resumption protocol specifically tailored for seamless 5G deployment. Our approach introduces three key innovations that distinguish it from existing solutions:

(1) Replay prevention: Unlike Bloom filter approaches [29], which can introduce false positives, or PPRF-based solutions [18] that require protocol modifications, we implement a server-side cryptographic replay cache combined with identity-bound session tickets. This cache allows for $\mathcal{O}(1)$ replay detection before expensive cryptographic operations, while the encrypted ticket enables cryptographic validation of client identity during session resumption.

(2) PFS with PQC: We enforce per-session STEK rotation using Kyber-768, ensuring that the compromise of session i does not jeopardize session $(i-1)$. Each resumption generates fresh quantum-resistant cryptographic material, effectively addressing the PFS gap present in standard TLS 1.3 resumption.

(3) Backward Compatibility: By strategically leveraging TLS 1.3 extension mechanisms and utilizing empty protocol functions, our design maintains full protocol compatibility. This allows for immediate deployment within existing 5G infrastructures without necessitating updates to currently deployed IoT devices. Through this architecture, we successfully eliminate replay attack vectors while retaining the performance advantages of session resumption.

B. Threat Model

Following the Dolev-Yao threat model [33], the adversary is assumed to be powerful, having complete control over the communication channel. We assume the adversary can intercept all messages between the UE and the gNB, allowing them to modify, inject, delay, reorder, or drop messages. However, they cannot break cryptographic primitives or access private keys unless compromised, despite having full knowledge of protocol specifications and access to all public keys.

In addition, we consider the following attacks. **Replay Attacks** involve an attacker capturing valid 0-RTT data and replaying it to the server, potentially causing duplicate operations or unauthorized access. **Man-in-the-Middle (MiTM) Attacks** occur due to the absence of immediate server-side verification in TLS 0-RTT, where an attacker can intercept or manipulate initial authentication messages. **Identity Spoofing Attacks** involve attackers impersonating legitimate devices by forging identity credentials. **Denial-of-Service (DoS) Attacks** involve flooding the server with ClientHellos containing distinct timestamps to exhaust replay cache capacity and computational resources. **Quantum Computing Attacks** represent a future threat where quantum computers could break traditional public key cryptography (RSA, ECC) using Shor's algorithm, compromising the security of TLS connections.

C. PQ-Enabled TLS Resumption Architecture and Approach

To enable PQC and optimize the re-authentication process, we leverage the core architecture of TLS 1.3 session resumption [16] (Figure 2) and design a new PQ-enabled TLS session resumption protocol as illustrated in Figure 3. Compared to standard TLS resumption, this new protocol has two critical additions: 1) **Server Validation** (replay cache checking and custom extension verification) prevents replay attacks and binds sessions to client identities; and 2) **Ticket Management** (Kyber-768-based STEK rotation and custom data embedding) ensures quantum resistance and PFS.

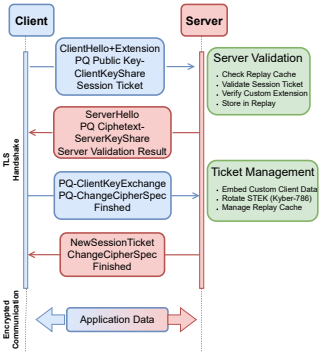

Fig. 3: PQ-Enabled Session Resumption

Our approach incorporates PQC, replay protection, and PFS to effectively address the security challenges faced by resource-constrained IoT devices. Figure 4 presents the flowchart of our approach, which is detailed below:

1) Step 1: Initial Client Request: Client sends $ClientHello$ with extension $e = (client\_id, nonce, timestamp)$ and session ticket (if available).

2) Step 2: Resume Session Check: Server analyzes incoming *ClientHello* for *valid session ticket* and custom *security extensions*. If no ticket $\rightarrow$ full TLS handshake; if ticket present

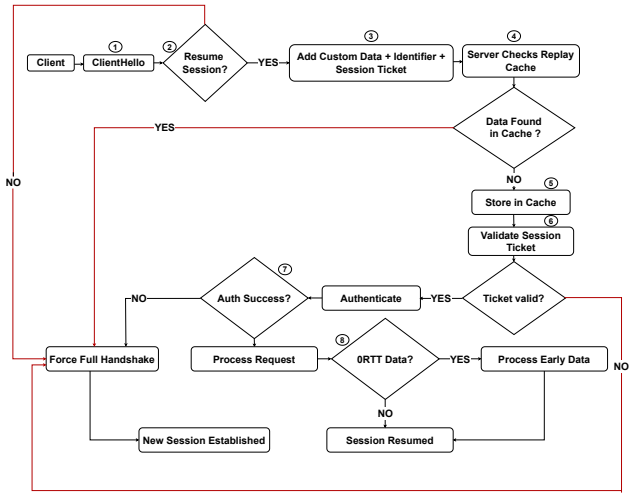

Fig. 4: Proposed Approach for Session Resumption

→ proceed with resumption steps. Uses *SSL_set_session()* to validate ticket presence and establish session context.

3) Step 3: Add Custom Data: If session ticket is present, extension data including *client_id*, high-entropy *random number*, and *timestamp* are embedded in *ClientHello*. Functions: **generate_client_ext_data** generates extension data, **client_ext_add_cb** adds to *ClientHello*, **custom_ext_cb** validates on server side, **new_session_ticket_cb** triggered when client receives new ticket.

4) Step 4: Replay Cache Validation: Primary defense against replay attacks. Server maintains **replay cache** (in-memory hash table with timestamp-based expiration), storing recently encountered extension values. Server extracts custom extension data from the session ticket and searches the cache. If found → *Force Full Handshake*; if not → *Store in Cache*. Key functions: *init_replay_cache()* initializes cache, *is_replay()* checks duplicates, *load_ext_data_from_file()* retrieves stored data, *SSL_set_session()* establishes resumption, *clean_replay_cache()* removes expired entries.

5) Step 5: Cache Storage: The replay cache operates as follows: First, the server hashes *(client_id, random number, timestamp)* from the ClientHello. It then checks for duplicates in the cache, and if none exist, stores the hash with its timestamp (±60-second clock skew). The cache maintains 1,000 entries maximum with a 300-second TTL, and expired entries are removed based on TTL. Additionally, to prevent cache pollution from invalid ClientHello, cache entries are added only after successful ticket validation. Furthermore, our approach does not introduce additional DoS vulnerabilities compared to standard TLS, as attackers can still flood servers with invalid certificates during full handshakes. However, ticket validation is computationally less intensive than certificate validation, making session resumption more DoS-resistant than full PQ handshakes (Table I).

6) Step 6: Session Ticket Validation: Decrypt ticket using quantum-resistant STEK (Kyber-768). Verify expiration, server issuance, and session state. If invalid → full handshake; else → Step 7. Functions: $load\_ticket\_keys()$, $save\_ticket\_keys()$.

7) Step 7: Authentication Success Check: Decrypt ticket to extract (RMS_{i-1}, e'_{i-1}). Verify: (1) $e_i = e'_{i-1}$, (2) timestamp within acceptable window Δ, (3) $client\_id_i = client\_id'_{i-1}$, (4) STEK decryption succeeds. If valid → Step 8; else → full handshake.

8) Step 8: Early Data Processing: If 0-RTT present: buffer via $SSL\_write\_early\_data()$ and process after validation using $early\_data\_cb()$. Else: proceed to $Session\ Resumed$.

D. Quantum-Resistant Key Management

After establishing the session resumption and replay protection mechanisms, the next critical component addresses the quantum threat landscape. The management of cryptographic keys in our session resumption process must ensure post-quantum security guarantees. Our key innovations are per-session STEK rotation using Kyber-768 for quantum resistance and identity binding through custom data embedded in quantum-resistant encrypted session tickets. Figure 5 presents our innovative key derivation workflow that integrates PQC while maintaining TLS 1.3 compatibility. The black arrows and white boxes represent the traditional TLS 1.3 handshake process, which includes the initial full handshake featuring Elliptic Curve Diffie-Hellman (ECDHE) key exchange, early secret derivation, and the creation of handshake and application traffic secrets. The workflow follows the standard HKDF-Extract operations to derive master secrets and resumption keys for subsequent sessions. All other functions, except for Session Ticket Validation and STEK, are derived from the standard TLS Key Generation and Session Resumption Mechanism outlined in Section II-C. At the same time, the red arrows and green boxes show our custom modifications that enhance security through post-quantum techniques and multi-layered validation. The explanations of the custom implementations are detailed below.

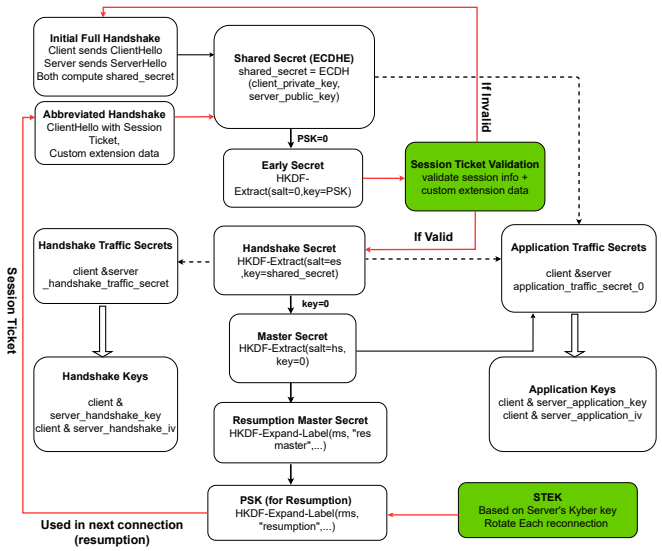

Fig. 5: Proposed TLS-Key Derivation Workflow

1) Custom Session Ticket Validation: Our workflow process verifies session information and custom extension data

during ticket validation. It uses *ssl_tlsext_ticket_key_cb* (standard OpenSSL callback) for encryption and decryption, *find_key_by_name* (custom function) to locate keys by their custom identifiers, and *create_new_key* (custom function) to generate tickets based on the extensions. The *save_ticket_keys* and *load_ticket_keys* (custom functions) handle the saving and restoring of keys after a server restart.

2) Quantum-Resistant STEK Implementation: We encrypt the session ticket with a quantum-enabled STEK key using the Kyber-768 standard. The STEK, based on the server's Kyber key, rotates with each reconnection for post-quantum security (shown in the green "STEK" box in Figure 5). Custom function *init_ticket_keys* initializes key management, with key rotation occurring every second for enhanced security.

3) Forced Fresh Session Ticket Generation: The server invokes the *ssl_tlsext_ticket_key_cb* (standard OpenSSL callback) during "Session Ticket Validation" phase in Figure 5, requiring a new ticket with a fresh key for each reconnection. This ensures unique cryptographic material for every connection, while the *create_new_key* function generates new keys based on custom extensions.

4) Enhanced Session Resumption Process: Client sends $ClientHello$ with session ticket and custom extension data. Validation failure triggers a full handshake. Functions *find_key_by_name*, *save_ticket_keys*, and *load_ticket_keys* (custom functions) ensure key continuity across server restarts.

E. Session Ticket Renewal and Management

Each session ticket is encrypted using a key derived from a Kyber-768 encapsulation operation, which is only known to the server and rotated for each resumption. This ensures that even when traditional certificates are used by the client, the session ticket is protected against quantum threats.

V. Security Analysis

This section examines the security features of the proposed PQ-Enhanced TLS Resumption mechanism in relation to the threat model in Section IV-B. To formally analyze the security of our enhanced TLS 1.3 session resumption protocol, we define an adversarial game denoted as $\mathcal{G}_{0-RTT-SR}$, $\mathcal{O}_{\text{Dec}}(t)$ is a Decryption Oracle that allows adversary to attempt session resumption using ticket t, simulating a client presenting a resumption ticket to the server. $\mathcal{O}_{\text{Test}}(t)$ is a Test Oracle that challenges the adversary to distinguish between the real session key and a random key for the session associated with ticket t. $\mathcal{O}_{\text{Corr}}$ is Corruption oracle that reveals the internal key material k_i for session i, modeling key compromise scenarios.

- If an adversary $\mathcal{A}$ queries $\mathcal{O}_{\text{Dec}}(t)$ (which represents a resumption attempt with ticket t) after successfully executing either an $\mathcal{O}_{\text{Test}}(t)$ or $\mathcal{O}_{\text{Dec}}(t)$ query, this should result in failure ($\perp$).
- In the game $\mathcal{G}_{0-RTT-SR}$, if $\mathcal{A}$ queries $\mathcal{O}_{\text{Corr}}$ after a $\mathcal{O}_{\text{Test}}(t)$ query (for session i), the revealed key $k_{\text{i,}}$ should not allow $\mathcal{A}$ to do the following:
  - Distinguish the real session key s_i (from RMS_{i-1}) from a random
  - Compute RMS_j for any $j < i$

Replay and MiTM Attack Resilience. We now formally analyze our protocol's resistance to replay and MiTM attacks.

Lemma 1 (Replay and MiTM Attack Resilience)**.** *For a correct anti-replay system with cache C and Validation function V, let $C' = C \cup \{e\}$ after processing the extension data e. Then, it follows that $V(C', e) = \perp$.*

Proof. For the anti-replay system with cache C, let $e = (client\_id, nonce, timestamp)$ be the extension data. After processing a legitimate resumption at time τ_i, we update $C' = C \cup \{h(e)\}$ where $h = SHA256$. For any subsequent query with the same extension data e at time $\tau' > \tau_i$, the validation function V computes $h' = h(e)$ and checks if $h' \in C'$. Since $h(e) \in C'$ by construction and SHA-256 is collision-resistant, $V(C', e) = \perp$ with overwhelming probability.

Theorem 1 (Replay and MiTM Resilience)**.** *Assuming the anti-replay cache and timestamp validation functions are correct, the protocol provides replay resilience against adversaries attempting to replay a previously used ticket t_{i-1} (which contains identifier e_{i-1}).*

Consider a legitimate session resumption i using ticket t_{i-1} with e_{i-1} at time τ_i. The Server computes:

$$is\_replay_{\text{e}_{i-1}} \rightarrow \text{adds}(k_{\text{e}_{i-1}}) \quad to\_replay\_cache$$

If an adversary attempts to replay t_{i-1} with e_{i-1} at a later time $\tau' > \tau_i$, the validation will fail:

$$is\_replay_{\text{e}_{i-1}} = \perp \quad \text{(by Lemma 1)}$$

As a result, the resumption attempt will be unsuccessful. Thus, even if an adversary manages to acquire a legitimate session ticket, both MiTM and replay attacks can be effectively prevented. This also indicates that an attacker cannot execute identity spoofing attacks, as they would need to regenerate a custom data extension successfully.

PFS: Next, we analyze the PFS properties of our session ticket key rotation system. A session ticket key rotation system is classified as forward-secure if the standard key security properties are upheld, even in scenarios where an adversary gains access to the current key k' after following a challenge query on an input x.

Theorem 2. *Assuming the ticket key rotation system exhibits forward security, and the used ephemeral key exchange (Diffie-Hellman (DH)/Kyber-768) provides forward secrecy. Under these assumptions, the protocol successfully achieves PFS for:*

- *Previous resumption secrets ($RMS_j, j < i$)*
- *Previous session keys ($tk_{C,j}, tk_{S,j}, j < i$)*

Assuming a compromise at time τ_c during which an adversary obtains k_{i}:

1. **Previous RMS**: For any $j < i$, the key $k_{\text{PPRF},j}$ has been discarded. The forward security of Kyber-768 prevents recovery of RMS_j.

2. **Previous Session Keys**: These keys are derived from RMS_{j-1} and ephemeral DH_j as:

$$\text{tk}_{C,j}, \text{tk}_{S,j} = f(\text{RMS}_{j-1}, \text{DH}_j)$$

The forward secrecy characteristics of DH/KEM ensure the protection of DH_j even after compromise [34].

Post-Quantum Security: Finally, we demonstrate that our protocol maintains security even against quantum adversaries.

Lemma 2 (PQ Composition)**.** *Assuming that the constructs of key rotation, OQS KEM, and signature schemes are founded on post-quantum assumptions, the protocol resists quantum adversaries.*

Key Rotation ensures that each reconnection generates a new session ticket with fresh quantum-resistant keys, preventing "store now, decrypt later" attacks. It uses robust AES-256/HMAC-SHA256, not vulnerable to known quantum attacks. **KEM**: Kyber-768, or other NIST PQC standards. **Signatures**: Dilithium2, Falcon, or SPHINCS+, a NIST PQC standard.

VI. Performance Evaluation

This section presents the experimental setup, metrics, and results comparing our PQ-enabled TLS protocol with traditional TLS in a 5G.

A. Setting up the 5G Testbed Environment

The performance evaluation was conducted by creating a physical testbed that simulates core 5G components distributed across four servers (three Laptops and a PC) and interconnected via Wi-Fi and VPN tunnels with the addition of our custom session resumption extension. We selected TLS tunneling for our verification method because it provides a standardized, interoperable mechanism [35] that can be seamlessly integrated into existing 5G infrastructure without proprietary modifications [11]. A similar setup was hosted on Google Cloud with NRF and CP located in Iowa, and gNB and UE in South Carolina, as shown in Figure 6.

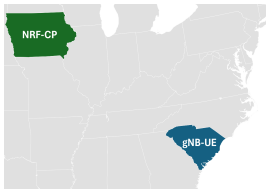

Fig. 6: Cloud-Setup

Hardware: Dell Precision Workstation (Intel i7, 16GB RAM) hosts *NRF. UE, gNB, and CP* machines hosted on three identical laptops (Intel i5, 8GB RAM, Ubuntu 20.04) and a Raspberry PI 4 (RPI). **Software Distribution and Connectivity:** Open5GS (5G core) [36], UERANSIM (UE/gNB) [37]. Machines interconnected via a Wi-Fi hotspot, simulating (UE->gNB->CP->NRF) with distinct TLS tunnels. OQS-OpenSSL library [38] used for generating PQ certificates, which enables PQ-TLS; OpenSSL [39] functions for custom session resumption. **Cryptographic Schemes for TLS Handshake**: All PQC operations (Falcon-512, Dilithium2, SPHINCS+-SHA256 signatures and Kyber-768 KEM) were implemented in C using the liboqs library integrated with OQS-OpenSSL fork [38], with all cryptographic operations executed in real-time on the physical hardware without simulation.

TLS Session Ticket Protection: Kyber-768 KEM used for encrypting ticket combined with AES-256-CBC and HMAC-SHA256 for encryption and integrity.

B. Metrics and Benchmarks

We evaluate three metrics comparing the initial full handshake (with certificates) with our custom resumption using a session ticket.

End-to-End Latency: Latency overhead from the UE, initiating the TLS connection until the UE receives the registration confirmation response from the NRF. The following delays were considered:

- *Initial TLS Handshake + Registration latency*: The total latency for the full TLS handshake and the UE registration message exchange.
- *Session Resumption + Registration Latency*: The duration for the abbreviated TLS handshake and the UE registration message exchange.

Message Overhead: The total number and size of the packets exchanged during UE registration with full and abbreviated handshakes. **Energy Consumption**: Energy consumption was measured on the UE (RPI acting as the UE/TLS Client) during the combined TLS and registration process for both initial and resumption scenarios. The same processes were run on the RPI to normalize the measured energy consumption.

C. Performance Results

This section provides a performance evaluation of UE registration overhead across local and cloud testbeds.

1) UE-registration in a Local Testbed: Our experiments are tailored towards UE registration and re-registration to the 5G core. Table I shows the initial experiments in the local testbed. As expected, the configuration without TLS authentication had the lowest delay at 0.365 seconds, with a message size of 13kB and a total of 30 messages.

TABLE I: Local Testbed Performance Comparison

ID	Certificate	Handshake Type	Delay	Size	No of msgs
1	noTLS	-	0.365	13kB	30
2	ECDSA	Full	0.401	21kB	43
		Our Approach	0.38	20kB	43
3	RSA	Full	0.417	25KB	43.3
		Our Approach	0.38	21.9kB	43.3
4	Fal512	Full	0.438	29kB	47
		Our Approach	0.38	22kB	43
5	Dil2	Full	0.592	32kB	50.9
		Our Approach	0.38	26kB	43
6	S+SHA	Full	1.09	67kB	75
		Our Approach	0.38	35.5kB	62

For PQ implementations, we achieve efficient handshake completion times of 0.438 seconds using Falcon512 and 0.592 seconds with Dilithium2. Additionally, in the S+SHA setup, we achieve 2.9× faster handshakes, 47% less bandwidth (35.5 kB vs 67 kB), and 17% fewer messages (62 vs 75). These consistent improvements across all certificate types demonstrate our protocol's superior efficiency within the framework of PQC while maintaining enhanced security guarantees.

2) UE-registration in Cloud testbed: Table II presents the results for the cloud testbed. It shows that both ECDSA and RSA performed similarly, with ECDSA full handshakes having an 8% longer delay and requiring 42 % larger data size. Both maintained 37 messages, reducing latency and data overhead. Falcon512, Dil2, and S+SHA full handshakes had more pronounced differences.

Note that larger digital signatures produced by the Falcon512, Dilithium 2, and S+SHA algorithms demonstrate enhanced performance in cloud environments due to better latency and CPU utilization, despite increased propagation delays. In contrast, smaller digital certificates perform more effectively in local settings.

TABLE II: Cloud Testbed Performance Comparison

ID	Certificate	Handshake Type	Delay	Size	No of msgs
1	noTLS	-	0.521	2kB	21
2	ECDSA	Full	0.611	4.4kB	37
3		Our Approach	0.566	3.1kB	37
4	RSA	Full	0.624	4.44kB	37
5		Our Approach	0.566	3.1kB	37
6	Fal512	Full	0.642	4.8kB	41
7		Our Approach	0.57	3.1kB	37
8	Dil2	Full	0.653	12.4kB	45
9		Our Approach	0.571	3.14kB	37
10	S+SHA	Full	0.686	39kB	69
11		Our Approach	0.582	3.3kB	56

3) Energy consumption Comparison of regular TLS with TLS session resumption: Across all certificate types, TLS session resumption demonstrates enhanced energy efficiency compared to full TLS handshakes, as shown in Table III. For ECDSA and RSA, the energy savings resulting from resumption are relatively modest, while Falcon512, Dilithium 2, and S+SHA exhibit more significant energy savings with resumption. S+SHA shows the most pronounced improvement, dropping from 0.4881 to 0.412 units (nearly a 16% reduction). This indicates a greater potential for energy savings when these resource-intensive operations are bypassed through session resumption. We note that we did not include the cloud setup results because they were similar to the local setup results.

TABLE III: Energy Consumption Comparison (Watts)

ID	**Certificate**	**TLS Full Handshake**	**Our Approach**
1	ECDSA	0.409	0.4
2	RSA	0.4393	0.4
3	Fal512	0.4738	0.409
4	Dil2	0.4794	0.41
5	S+SHA	0.4881	0.412

For instance, an IoT device powered by a 6Wh battery (e.g., two AA batteries) that re-authenticates 48 times daily can achieve an extended operational life of about 303.4 days with our approach, compared to about 256.1 days with a full handshake, effectively saving around 47.3 days of battery life per device before requiring replacement.

Our approach enhances performance (latency, data size, messages, energy) across all certificates while providing robust security. This is achieved through a novel custom resumption that utilizes empty bits of the TLS handshake procedure, alongside improvements to the TLS key generation process.

D. Discussion

PQC is crucial for securing communication protocols against quantum threats but adds performance challenges. The experimental results demonstrate that our approach of TLS session resumption serves as a vital mitigation strategy, effectively transforming these initial overheads into manageable one-time costs. The proposed method significantly enhances performance for efficient communication during frequent re-authentication.

These performance improvements are achieved while simultaneously addressing critical security vulnerabilities in standard TLS 1.3 session resumption. Our protocol protects against replay attacks, ensures PFS with per-session STEK rotation, and binds sessions to client identities, all while using quantum-resistant cryptography

These enhancements scale effectively across thousands of IoT devices, addressing control plane bottlenecks in 5G networks. With 10,000 IoT devices re-authenticating 48 times daily, S+SHA session resumption eliminates 6.24 million signalling messages and reduces data transfer by 15.12 GB per day. For high-frequency re-authentication (every 5-10 minutes), processing time savings exceed 500,000 seconds daily. For Quality-of-Service critical applications, such as Vehicle-to-Network (V2N) [40], our approach provides consistent latency and energy profiles.

Our solution offers broad compatibility with several PQ algorithms, allowing network operators to select cryptographic methods while optimizing performance. Consequently, our enhanced PQ-enabled TLS session resumption becomes a key enabler for the practical deployment of PQC in resource-constrained, high-density 5G core networks. Our contribution achieves two primary outcomes:

I. **PQ-Enabled Efficiency**: We enable integration of PQC through novel session resumption, making it feasible in performance-critical settings.

II. **Universal Security Enhancement**: In addition to the advantages of PQC, we significantly strengthen the traditional session resumption protocol by addressing inherent vulnerabilities and enhancing security mechanisms.

VII. Conclusion

This paper successfully addresses the challenges of integrating PQC securely into TLS for 5G IoT devices while optimizing performance. We introduced a novel approach for IoT UE re-authentication that secures TLS session resumption by eliminating its vulnerability to replay attacks and ensuring PFS. This secure session resumption mechanism minimizes PQC overhead in 5G networks by performing expensive cryptographic operations only during initial handshake, not during frequent re-authentication events. Our solution achieved the security enhancement for re-authentication procedure without requiring any changes to existing TLS client characteristics, promoting ease of deployment. The practicality and effectiveness of our methodology are validated by experimental results obtained from both local and cloud-based 5G testbeds. This work demonstrates the feasibility of effectively employing PQC within TLS session resumption, provided that the security vulnerabilities of standard extensions are appropriately addressed.

Acknowledgment

This research was funded by the US NSF under grant No. 2147196.

REFERENCES

[1] I. Analytics, "Number of connected iot devices in 2023: Nearing 16 billion," https://iot-analytics.com/number-connected-iot-devices/, 2023, accessed: 2024-10-23.

[2] ETSI, "3GPP TS 33.501 V15.1.0 - Security architecture and procedures for 5G system (Release 15)," European Telecommunications Standards Institute (ETSI), Tech. Rep., 2018.

[3] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, and V. Stettler, "A formal analysis of 5g authentication," in Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, 2018, pp. 1383–1396.

[4] A. Koutsos, "The 5g-aka authentication protocol privacy," in 2019 IEEE European symposium on security and privacy (EuroS&P). IEEE, 2019, pp. 464–479.

[5] C. Cremers and M. Dehnel-Wild, "Component-based formal analysis of 5g-aka: Channel assumptions and session confusion," in Network and Distributed System Security Symposium (NDSS). Internet Society, 2019.

[6] R. P. Jover and V. Marojevic, "Security and protocol exploit analysis of the 5g specifications," IEEE Access, vol. 7, pp. 24 956–24 963, 2019.

[7] C. Ugwuishiwu, U. Orji, C. Ugwu, and C. Asogwa, "An overview of quantum cryptography and shor's algorithm," Int. J. Adv. Trends Comput. Sci. Eng, vol. 9, no. 5, 2020.

[8] D. Stebila and M. Mosca, "Post-quantum key exchange for the internet and the open quantum safe project," in International Conference on Selected Areas in Cryptography. Springer, 2016, pp. 14–37.

[9] National Institute of Standards & Technology, "Post-quantum cryptography," Jul. 22, 2022. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography

[10] K. Sabanci, "Exploring post-quantum cryptographic schemes for tls in 5g nb-iot: Feasibility and recommendations," Master's thesis, Marquette University, 2023.

[11] Y. Hanna, D. Pineda, M. Veksler, M. Paudel, K. Akkaya, M. Anastasova, and R. Azarderakhsh, "Integrating post-quantum tls into the control plane of 5g networks," in 2024 IEEE International Performance, Computing, and Communications Conference (IPCCC). IEEE, 2024, pp. 1–8.

[12] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis, "Post-quantum authentication in tls 1.3: A performance study," Cryptology ePrint Archive, 2020.

[13] C. Paquin, D. Stebila, and G. Tamvada, "Benchmarking post-quantum cryptography in tls," in International Conference on Post-Quantum Cryptography. Springer, 2020, pp. 72–91.

[14] 3rd Generation Partnership Project (3GPP), "Non-Access-Stratum (NAS) protocol for 5G System (5GS); Stage 3," 3rd Generation Partnership Project (3GPP), Technical Specification TS 24.501, sep 2020, version 16.5.1, Release 16.

[15] S. Rommer, P. Hedman, M. Olsson, L. Frid, S. Sultana, and C. Mulligan, 5G core networks: powering digitalization. Academic Press, 2019.

[16] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, 2018. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8446

[17] S. Hebrok, S. Nachtigall, M. Maehren, N. Erinola, R. Merget, J. Somorovsky, and J. Schwenk, "We really need to talk about session tickets: A {Large-Scale} analysis of cryptographic dangers with {TLS} session tickets," in 32nd USENIX Security Symposium (USENIX Security 23), 2023, pp. 4877–4894.

[18] N. Aviram, K. Gellert, and T. Jager, "Session resumption protocols and efficient forward security for tls 1.3 0-rtt," Journal of Cryptology, vol. 34, no. 3, p. 20, 2021.

[19] A. Fakhreddine, C. Bettstetter, S. Hayat, R. Muzaffar, and D. Emini, "Handover challenges for cellular-connected drones," in Proceedings of the 5th Workshop on Micro Aerial Vehicle Networks, Systems, and Applications, 2019, pp. 9–14.

[20] J. Salowey, H. Zhou, P. Eronen, and H. Tschofenig, "Transport layer security (tls) session resumption without server-side state," Tech. Rep., 2008.

[21] D. Joseph, R. Misoczki, M. Manzano, J. Tricot, F. D. Pinuaga, O. Lacombe, S. Leichenauer, J. Hidary, P. Venables, and R. Hansen, "Transitioning organizations to post-quantum cryptography," Nature, vol. 605, no. 7909, pp. 237–243, 2022.

[22] M. J. H. Faruk, S. Tahora, M. Tasnim, H. Shahriar, and N. Sakib, "A review of quantum cybersecurity: threats, risks and opportunities," in 2022 1st International Conference on AI in Cybersecurity (ICAIC). IEEE, 2022, pp. 1–8.

[23] National Institute of Standards & Technology, "Nist announces first four quantum-resistant cryptographic algorithms," Jul. 5, 2022. [Online]. Available: https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms

[24] M. Arapinis, L. Mancini, E. Ritter, M. Ryan, N. Golde, K. Redon, and R. Borgaonkar, "New privacy issues in mobile telephony: fix and verification," in Proceedings of the 2012 ACM conference on Computer and communications security, 2012, pp. 205–216.

[25] A. Braeken, M. Liyanage, P. Kumar, and J. Murphy, "Novel 5g authentication protocol to improve the resistance against active attacks and malicious serving networks," Ieee Access, vol. 7, pp. 64 040–64 052, 2019.

[26] T. Liu, G. Ramachandran, and R. Jurdak, "Post-quantum cryptography for internet of things: a survey on performance and optimization," arXiv preprint arXiv:2401.17538, 2024.

[27] Y. Hanna, J. Bozhko, S. Tonyali, R. Harrilal-Parchment, M. Cebe, and K. Akkaya, "A comprehensive and realistic performance evaluation of post-quantum security for consumer iot devices." Elsevier, 2025, p. to appear.

[28] J. Zhou, W. Fu, W. Hu, Z. Sun, T. He, and Z. Zhang, "Challenges and advances in analyzing tls 1.3-encrypted traffic: A comprehensive survey," Electronics, vol. 13, no. 20, p. 4000, 2024.

[29] D. Derler, K. Gellert, T. Jager, D. Slamanig, and C. Striecks, "Bloom filter encryption and applications to efficient forward-secret 0-rtt key exchange," Journal of Cryptology, vol. 34, pp. 1–59, 2021.

[30] F. Dallmeier, J. P. Drees, K. Gellert, T. Handirk, T. Jager, J. Klauke, S. Nachtigall, T. Renzelmann, and R. Wolf, "Forward-secure 0-rtt goes live: implementation and performance analysis in quic," in Cryptology and Network Security: 19th International Conference, CANS 2020, Vienna, Austria, December 14–16, 2020, Proceedings 19. Springer, 2020, pp. 211–231.

[31] M. Fischlin and F. Günther, "Multi-stage key exchange and the case of google's quic protocol," in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, 2014, pp. 1193–1204.

[32] K. Tange, S. Mödersheim, A. Lalos, X. Fafoutis, and N. Dragoni, "rtls: Secure and efficient tls session resumption for the internet of things," Sensors, vol. 21, no. 19, p. 6524, 2021.

[33] D. Dolev and A. Yao, "On the security of public key protocols," IEEE Transactions on information theory, vol. 29, no. 2, pp. 198–208, 1983.

[34] M. Cebe and K. Akkaya, "A replay attack-resistant 0-rtt key management scheme for low-bandwidth smart grid communications," in 2019 IEEE Global Communications Conference (GLOBECOM). IEEE, 2019, pp. 1–6.

[35] H. Akter, S. Jahan, S. Saha, R. H. Faisal, and S. Islam, "Evaluating performances of vpn tunneling protocols based on application service requirements," in Proceedings of the Third International Conference on Trends in Computational and Cognitive Engineering: TCCE 2021. Springer, 2022, pp. 433–444.

[36] S. Lee and Open5GS Contributors, "Open5gs," 2026, open source implementation of 5G Core and EPC. [Online]. Available: https://github.com/open5gs/open5gs

[37] A. Güngör, "Ueransim," 2026, open source 5G UE and RAN simulator for 5G Core Network. [Online]. Available: https://github.com/aligungr/UERANSIM

[38] "Oqs-openssl," Feb 2022. [Online]. Available: https://github.com/open-quantum-safe/openssl/blob/OQS-OpenSSL_1_1_1-stable/README.md

[39] OpenSSL Project, "OpenSSL: Cryptography and SSL/TLS Toolkit," https://github.com/openssl/openssl, 2025, accessed: 2025-05-07.

[40] 3GPP, "Architecture enhancements for 5G System (5GS) to support Vehicle-to-Everything (V2X) services; Stage 2 (Release 16)," 3rd Generation Partnership Project (3GPP), Technical Specification TS 23.287, 2020, version 16.2.0. [Online]. Available: https://www.3gpp.org/DynaReport/23287.htm