QUPKIOT: QUIC BASED P2P PUBLIC KEY INFRASTRUCTURE FOR IOT

Ozan Tarlan, Kübra Kalkan, Ilgın Şafak, and Hasan Sözer

Ozyegin University and University of Jyväskylä, Fibabanka R&D Center

ABSTRACT

We introduce a decentralized public key infrastructure (DPKI) for the Internet of Things (IoT), leveraging Kademlia-based Distributed Hash Tables (DHT) for robust authentication and elimination of single points of failure. Building on this DPKI foundation, we propose a modified Quick UDP Internet Connections (QUIC) protocol tailored for peer-to-peer IoT communication, aiming to reduce overhead and latency. This peer-to-peer QUIC framework ensures low-latency, zero round trip time(0-RTT) session resumption at the network's edge, making it suitable for resource-constrained devices. To further optimize performance, we employ Elliptic Curve Diffie-Hellman (ECDH) for secure handshakes and integrate Salsa20 as a lightweight symmetric encryption algorithm. Simulation results in NS-3 demonstrate faster secure session establishment times and more efficient cryptographic operations compared to existing solutions, highlighting the advantages of our approach for IoT environments.

KEYWORDS

QUIC, IoT, Networks Security, P2P

1. INTRODUCTION

Public Key Infrastructure (PKI) is a cornerstone of security for many modern systems, enabling secure communication, authentication, and data integrity. As digital ecosystems grow in complexity, the need for robust and scalable PKI solutions becomes increasingly critical. This is especially true for the Internet of Things (IoT), where the increase of interconnected devices demands secure and efficient mechanisms to manage identities and encryption keys.

The rapid expansion of IoT has transformed industries, with devices now deployed in diverse environments such as smart factories, homes, wearables, and financial services. These devices are integral to modern digital ecosystems, facilitating seamless communication and commerce. However, as IoT continues to scale, ensuring the security of billions of resource-constrained devices has become a significant challenge. Many of the IoT devices possess limited processing power, memory, and energy resources, making traditional security solutions often reliant on high-overhead encryption algorithms and centralized architectures impractical or inefficient[32].

Securing IoT with Decentralized Public Key Infrastructure (DPKI) offers a promising solution to these challenges. Unlike traditional PKI, which relies on centralized authorities, DPKI distributes trust across the network, eliminating single points of failure[31] and enhancing resilience. This decentralized approach is particularly beneficial for IoT, as it allows devices to authenticate and communicate securely without depending on a central server. By removing the need for intermediaries, DPKI reduces latency, lowers hardware costs, and mitigates the risk of bottlenecks or failures that could compromise the entire network.

DPKI is inherently synergistic with peer-to-peer (P2P) communication models, which further decentralize IoT networks. In a P2P model, messages travel directly between devices, bypassing

David C. Wyld et al. (Eds): CMLA, CSEIT, NLPD, GRAPH-HOC, SPM, NeTCoM, NCS, ArIT, DMSE – 2025
pp.193 -206, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151316

centralized brokers that introduce additional latency and potential vulnerabilities. However, the absence of a central authority in P2P systems creates unique challenges for node authentication and key management. To address this, Distributed Hash Tables (DHTs), such as Kademlia, can be used to implement DPKI. Each node in the network stores a portion of the overall data, enabling secure and decentralized key distribution and verification. This approach not only enhances scalability but also strengthens the network's resistance to denial-of-service (DoS) attacks [1].

To further optimize IoT security and performance, the QUIC protocol can be leveraged as a foundation for secure communication. Originally designed for web traffic, QUIC offers several advantages for IoT [33], including reduced latency, faster connection establishment, and support for 0-RTT resumption. By tailoring QUIC to IoT-specific needs such as integrating lightweight cryptographic suites devices can achieve efficient and secure data transfers without overwhelming their limited computational resources. This makes QUIC a cost-effective solution for IoT deployments, particularly in time-sensitive scenarios where responsiveness is critical.

Lightweight cryptographic algorithms are crucial for resource-constrained IoT devices. Traditional encryption can overwhelm these systems, compromising performance and energy efficiency. We propose using ECDH for key exchange and Salsa20 for symmetric encryption. ECDH delivers strong security with small key sizes, while Salsa20 is a lightweight alternative to AES especially where specialized hardware is lacking. Combined, they reduce overhead and power usage, making them well-suited for IoT environments. We chose Salsa20 for its low overhead on constrained IoT devices, but acknowledge that ChaCha20Poly1305, as standardized in QUIC/TLS 1.3, offers integrated AEAD for stronger security. While our protocol ensures integrity through signature-based authentication, future work may explore ChaCha20-Poly1305 for enhanced security if device capabilities allow.

In summary, the combination of DPKI, P2P communication, QUIC, and lightweight cryptographic suites addresses the key challenges of IoT security: scalability, efficiency, and resilience. This paper proposes a modified QUIC-based protocol that integrates ECDH, Salsa20, and a DHT-driven P2P model to create a robust, low-latency, and secure IoT communication framework.

The primary contributions of this paper are as follows:

- We introduce a Decentralized Public Key Infrastructure (DPKI) based on the Kademlia Distributed Hash Table (DHT) to enable secure and scalable node authentication without relying on centralized authorities. This approach eliminates single points of failure, enhances network resilience, and supports seamless scalability.
- We propose a peer-to-peer (P2P) IoT messaging system that leverages the QUIC protocol for efficient and secure communication, tailored specifically for resource-constrained devices.
- Our solution integrates lightweight cryptographic algorithms, including Salsa20 for symmetric encryption and ECDH for secure key exchange and handshake, ensuring robust security with minimal computational overhead.

Together, these contributions provide a comprehensive framework for secure, lowlatency, and decentralized IoT communication. The remainder of this paper is organized as follows: Section II provides related work and background on existing IoT communication strategies. Section III details the system model, the handshake protocol, and security analysis of our approach. Section IV presents the experimental setup and simulation results, comparing our protocol's performance against IoT-PKI [1]. Section V concludes the paper and discusses directions for future work. By combining specialized ciphers, a low-latency transport protocol, and decentralized authentication, our solution aims to advance the state of IoT security and efficiency, ultimately contributing to a more reliable and scalable IoT ecosystem.

2. RELATED WORKS

This section provides a brief summary of existing IoT protocols that use QUIC, related work focusing on QUIC in IoT and P2P communications, as well as P2P examples of public key authentication.

Table 1. Key Properties and Their Comparison Across Published Studies

No.	IoT	Decentralized	Ledger Type	Key Exchange	DHT?	Blockchain?	Certificates	Layers	Trust	DID
Ours	✓	✓	-	✓	✓	×	×	application-transport	×	×
1	✓	✓	Blockchain	✓	×	✓	✓	application	×	×
2	✓	✓	Smart Contracts	×	×	✓	✓	application	×	×
3	✓	✓	Blockchain	×	×	✓	✓	application	×	×
4	✓	✓	Smart Contracts	×	×	✓	×	application-network	×	×
5	✓	✓	emercoin/Smart Contracts	×	×	✓	✓	application	×	×
6	✓	✓	proof of membership BC	×	×	✓	×	application	✓	✓
7	✓	✓	Emercoin	×	×	✓	×	application	×	×
8	×	✓	Blockchain	×	×	✓	×	application	×	×
9	✓	✓	Algorand	×	×	✓	×	application-transport	×	×
10	✓	✓	blockchain	×	×	✓	✓	application-network	✓	×
11	✓	✓	Smart Contracts	×	×	✓	✓	application	×	×
12	×	✓	Sovrin	×	×	✓	✓	application	×	✓
13	✓	✓	Smart Contracts/Blockchain/IPFS	×	✓	✓	✓	application	×	×
14	×	✓	Blockchain	×	×	✓	✓	application	×	×
15	✓	✓	Blockchain	✓	×	✓	×	application	×	✓
16	✓	✓	-	✓	×	×	×	application	×	×
17	✓	×	-	✓	×	×	×	application	×	×

We conducted a literature review to find candidates for performance comparison. Table I lists the key properties of published studies to position our work with respect to these. Works [1] to [15] utilize blockchain-based ledgers to manage decentralized public key infrastructures (DPKI). These approaches leverage the immutable and distributed nature of blockchains to store keys and manage trust in IoT environments. [2], [4], [5], [11], and [13] emphasize the use of smart contracts to facilitate key exchange and management within IoT systems. [6], [7], [9], [12], and [16] explore alternative mechanisms for decentralized key management. These include proof-of-membership-based blockchains, emercoin, and specialized ledgers like Algorand or Sovrin. [6], [10], [12], and [15] incorporate features like Decentralized Identifiers (DIDs) and additional trust layers to strengthen IoT systems. DIDs provide a standardized method for secure, decentralized identity management, aligning with efforts to reduce reliance on centralized authorities. [16] and [17] focus on peer-to-peer (P2P) key exchange and authentication mechanisms for IoT systems without the use of DPKI. Many of the mentioned related works from the table use blockchain, we utilize the DHT in order to reduce the data stored in the IoT devices. Since the use of blockchain also stores expired/invalid data (in this context certificates and device identifiers) which may fill the memory of resource constrained IoT devices. We found IoT-PKI [1] most suitable for comparison due to its similarity of adopted concepts. They use decentralized public key infrastructure, decentralized network topology, authenticated key exchange for IoT. In [1], the authors propose IoT-PKI, a decentralized PKI framework for IoT that replaces traditional Certificate Authorities (CAs) with distributed nodes in a blockchain network. This design mitigates single points of failure in centralized PKI, particularly significant as IoT scales to billions of devices. Moreover, IoT-PKI empowers device owners to manage their own certificates, preventing manufacturers from gaining access to private keys. The feasibility and efficiency of the scheme are validated through a prototype implementation, demonstrating its potential to enhance large-scale IoT security.

In [18], the authors evaluate AMQP over QUIC, leveraging a standardized message broker architecture for reliable communication among heterogeneous software components, whereas our work takes a point-to-point approach without relying on central intermediaries. Similarly, while [19] integrates QUIC with MQTT to enhance performance in brokerbased IoT applications, we forgo a broker and design a P2P-friendly, QUIC-based protocol to remove single points of failure. Finally, [20] uses a proxy-based scheme for CoAP over QUIC to achieve multi-streaming efficiency, but our method eliminates proxies, adopts a distributed hash table (DHT) for decentralized authentication, and employs ECDH and Salsa20 to minimize overhead in low-power IoT devices. By customizing QUIC to these P2P requirements, we address security, latency, and single points of failure simultaneously. MQTT over QUIC still relies on centralized brokers, introducing single points of failure and latency bottlenecks. CoAP over QUIC uses proxy-based architectures to enhance streaming, yet this approach sacrifices full decentralization. In contrast, our design eliminates both brokers and proxies by adopting a fully peer-to-peer (P2P) communication model, secured through a DHT-based decentralized authentication mechanism. This setup enables robust security, low-latency 0-RTT QUIC session resumption, and reduced cryptographic overhead using ECDH and Salsa20 advantages not jointly realized by the above protocols. As a result, our protocol provides broader and more sustainable benefits for scalable, secure IoT deployments.

A novel approach is proposed in [21] for peer-to-peer Bitcoin transactions using the QUIC network protocol. Unlike the classic Bitcoin transactions, the proposed QUIC Bitcoin Transactions, leveraging the advantages of QUIC, provides a secure and fast exchange of unpublished transactions. Additionally, the introduction of QUIC Bitcoin Channels ensures encrypted payment channels, offering secure peer-to-peer transactions without the need for new hardware requirements. While reducing hardware requirements by leveraging P2P, and utilizing QUIC for transactions, they do not focus on IoT communications.

In [22], the authors introduce Certcoin, a decentralized PKI that leverages Bitcoin and Namecoin to address vulnerabilities in traditional PKIs (e.g., unauthorized identity re-registrations). In contrast, our work proposes a simpler, DHT-based approach for registering and authorizing clients via public keys—without using blockchain.

In papers [23], [24] Elliptic Curve Cyptography (ECC)-based authentication and communications schemes for IoT were discussed. Also the TLS 1.3 which is what the latest versions of QUIC, can utilize but is not limited to ECDH cipher [26]. [25] also discusses the use of salsa20 for lightweight IoT communications. We leverage the above-mentioned ciphers for lightweight IoT communications. Recent cryptanalysis of Salsa20 indicates that no practical exploit has been discovered, and the cipher remains secure against all known attacks. Meanwhile, the latest versions of both TLS (Transport Layer Security) and QUIC (Quick UDP Internet Connections) support elliptic curve Diffie-Hellman (ECDH) key exchange, reflecting industry-wide adoption of ECDH to enhance confidentiality and forward secrecy in secure communication protocols.

2.1. System Model

The model proposed in this paper assumes that the protocol is used to serve any capable device ranging from high complexity to lightweight IoT sensors. Devices can join and communicate anytime throughout the simulation. These devices generate network traffic after securing the session using the handshake protocol. A malicious device is assumed to be able to join and listen to any message in the network. However, the malicious device is assumed to have no authority to write any address-public key pairs to the DHT. The authority to write new entries is given to the nodes existing from the start of the simulation. The overview of the described contributions can be seen in Fig. 1.

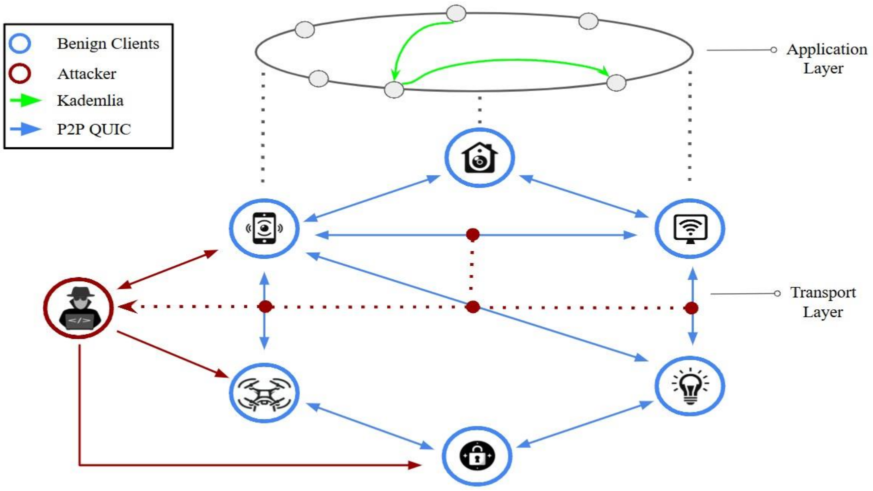


Fig.1. System and Adversary Model

3. Handshake Protocol

3.1. System Components

QUIC is a modern, secure transport protocol [27] that addresses evolving Internet reliability needs. Having undergone extensive real-world testing, it offers a robust foundation for IoT scenarios. Its UDP underpinnings simplify NAT and firewall traversal while supporting diverse IoT environments. QUIC also integrates congestion control inspired by TCP for adaptive data transfer. The incorporation of TLS 1.3 further solidifies its commitment to low-overhead security, protecting IoT privacy and integrity.

QUIC's rapid connection setup, efficient multiplexing, robust security, and advanced traffic management are well-suited for IoT, improving network performance and safety. In this paper, we adapt QUIC to meet decentralized IoT requirements with a tailored handshake protocol.

A decentralized IoT approach reduces unnecessary data paths and centralization costs, alleviating a core limitation of conventional IoT protocols. Solutions like MQTT rely on intermediaries, but our direct P2P method removes latency bottlenecks (see Fig. 2) and single points of failure. However, decentralization creates challenges in node authentication, traditionally handled by a central authority. To address this, we employ a Distributed Hash Table (DHT) as a decentralized Public Key Infrastructure (PKI). Specifically, we use Kademlia DHT for storing and retrieving public keys, enabling a scalable, robust peer-topeer topology. Although we did not include empirical measurements, Kademlia's scalability is well-established analytically. It offers O(log*n*) lookup complexity and O(*k* log*n*) routing table maintenance, where *n* is the number of nodes and *k* is the bucket size. These properties ensure efficient operation even under churn, making it suitable for large-scale, dynamic IoT networks. Future work will include empirical validation of these performance bounds.

Given the low-power nature of IoT devices, our protocol employs Elliptic Curve DiffieHellman (ECDH) for key exchanges, providing efficient security, and adopts Salsa20 for symmetric encryption to minimize overhead. To address the reduced resilience associated with ECDH and Salsa20, frequent ephemeral key rotations using fresh ECDH exchanges could be used to ensure forward secrecy and limit the exposure time of session keys.

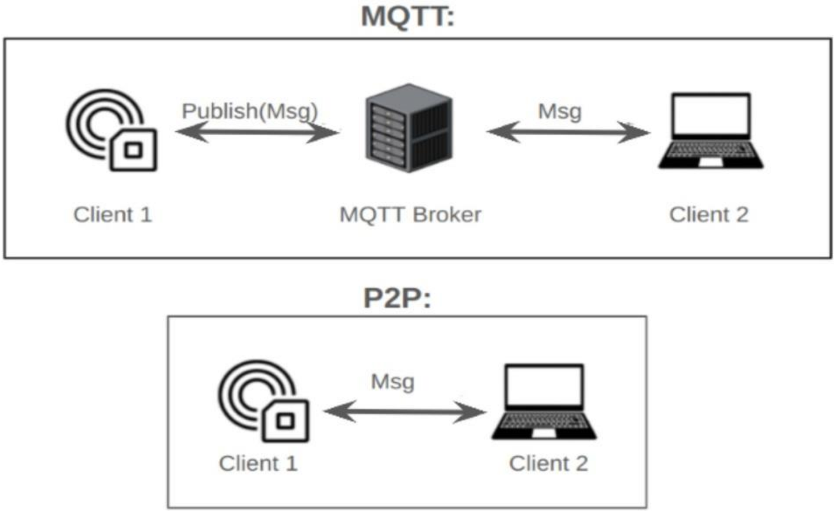


Fig.2. Message flow in MQTT vs P2P

Additionally, employing key derivation functions (KDFs) with strong cryptographic hashes alongside periodic re-keying can be used in mitigating potential cryptanalytic risks, as well as enforcing short-lived sessions to enhance security. Although it is out of the scope of this paper, these measures could be complemented with continuous monitoring for anomalies. By combining these elements, our solution delivers lightweight, secure, authenticated, and decentralized IoT communication. We modify the QUIC protocol in the transport layer and utilize Kademlia DHT in the application layer to ensure seamless key management and authentication.

P2P architectures require robust security due to increased direct interactions among nodes, increasing the risk of threats such as impersonation, man-in-the-middle, and Sybil attacks. By implementing an optimized handshake protocol that leverages lightweight elliptic curve cryptography (ECC), digitally signed key entries stored in the DHT, immediate rejection of invalid or suspicious keys, and mutual authentication using ephemeral keys, these vulnerabilities are effectively mitigated without significantly impacting network performance, ensuring secure, trusted communication in resource-constrained P2P environments. To ensure Sybil resistance, only genesis nodes have write access to the DHT, and all address–public key mappings are digitally signed. Malicious nodes may join and communicate but cannot inject forged identities, as unauthorized entries are rejected during handshake verification. This cryptographic restriction prevents identity spoofing and limits Sybil attacks.

3.2. Adversary Model

This paper adopts the Dolev-Yao model outlined in [29], wherein the adversary is capable of eavesdropping, replaying, and fabricating messages within the network. Eavesdropping is the unauthorized interception and listening of transmitted network messages. This passive attack involves an attacker secretly monitoring and capturing messages in the network as it is being sent from one device to another, without changing the message. In a replay attack, an attacker captures legitimate network packets, like a login attempt or message, and sends it again to mimic the initial activity. Such attacks take advantage of reusing valid authentication messages, enabling the attacker to gain unauthorized actions or access without direct knowledge of the data involved. Message fabricating involves creating and sending messages that appear to be from a legitimate source. This type of attack aims to deceive the recipient device into believing that the message is authentic, potentially leading to unauthorized access, or other malicious outcomes. In Fig. 1. we can see an example of the network setup. Blue circles indicate the benign devices while red circle displays an attacker. Straight arrows display the connected devices, an attacker can also connect to any device. Any communication in the network can be subject to eavesdropping as shown with the dotted red lines and any communication can be replayed and forged indicated with red lines. However, the attacker or any newly joined devices do not have a write access in the DHT. The objectives of the attacker are the following:

- Generating or acquiring the key utilized in the session.
- Gaining access to device information.
- Obtaining confidential information exchanged during communication.

3.3. Handshake Protocol Description

This section presents the details of the outlined handshake protocol. The protocol enables IoT devices to establish a secure communication session by employing both asymmetric encryption using elliptic curve cryptography and symmetric encryption using Salsa20. Devices can securely generate a unique session key, ensuring end-to-end encryption and secure communication.

In the authenticated handshake protocol (see Fig. 3), a client that wants to initiate the handshake must first obtain the recipient’s public key from the DHT. The initiating client is assumed to know the recipients identifier (e.g., e-mail address, phone number, etc.), which it uses to look up its public key on the DHT. The client initiates the connection by sending a hello message with its public key, and a hexadecimal nonce of length 16 encrypted with the receivers public key. The presented handshake protocol provides enhanced security over its counterparts by sending encrypted hello messages. The receiver decrypts the message with its private key. The receiver then sends its public key XOR’ed with the sender’s public key and the initialization vector IV encrypted with the senders public key. Both clients use the XOR function to generate the same initialization vector (IV). With the public keys, clients agree on the shared secret and they send each other. The agree() function refers to the key agreement step using Elliptic Curve Diffie-Hellman (ECDH). Using the Salsa20 cipher, clients encrypt the nonces they received with the shared secret and IV. Clients validate the received encrypted message by decrypting it with the shared secret and IV. If the decrypted message is equal to their own nonce value that is sent to this specific client, they send a symmetrically encrypted finished message verifying that the handshake is complete. This entry is sent encrypted using the DHT's public key. With this authenticated handshake protocol, we guarantee that clients are who they claim to be.

Table 2. Definition and Acronyms

Acronym	Definition
Kp1	Client 1 Public Key
Kp2	Client 2 Public Key
Ks	Shared Secret
Cn1	Client 1 Nonce
Cn2	Client 2 Nonce
IV	Initialization Vector
Addr1	Client 1 Address
Addr2	Client 2 Address

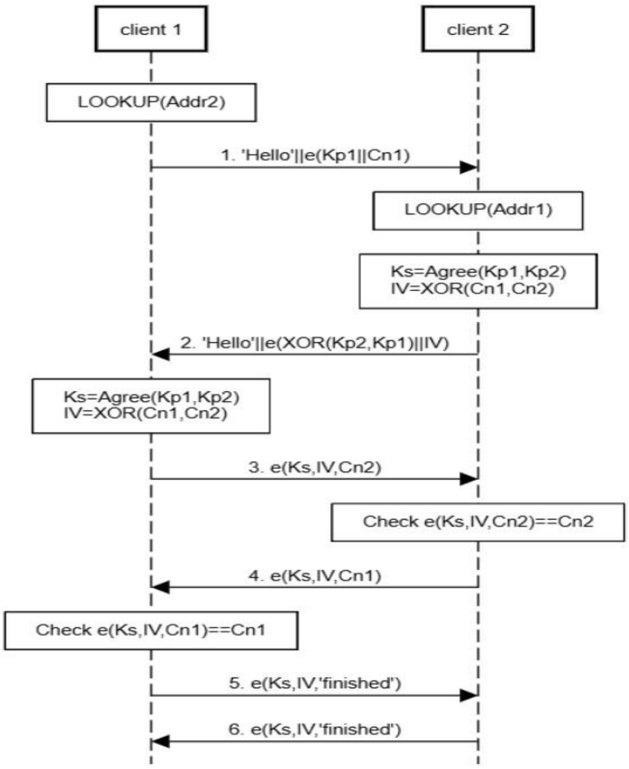


Fig.3. End-to-end Encrypted Handshake Protocol

Related definitions of the acronyms such as keys, nonces, and addresses can be found in Table 1. The function e() is not included in this table due to its long definition. The function e() in the sequence diagram presents the encryption operations. This function e takes 2 parameters for asymmetric encryption and 3 parameters for the symmetric encryption. Asymmetric encryption parameters are; the key and the message. The function encrypts the second parameter (the message to be encrypted) with the first parameter (encryption key). Symmetric encryption parameters are the key, IV, and the message. In this case, the third parameter(message to be encrypted) is encrypted based on the first two parameters; key and IV. Once the handshake protocol is successfully completed, the shared secret, Ks, agreed upon is used henceforth to establish secure peer-to-peer communications between the devices. The shared secret could be periodically updated to maintain security, with the frequency depending on the specific security requirements and policies in place.

3.4. Security Analysis

The security analysis of the suggested protocol is detailed below, together with an analysis of the likelihood of an attacker compromising the security assurances to fulfill malicious objectives. This evaluation is grounded in several assumptions: Firstly, the security of the signature scheme employed by protocol participants and the DHT is secure, making it infeasible for an attacker to forge a signature without access to the private key. Secondly, both of the client nonces are randomly selected twice with negligible probability of being the same. Thirdly, the session key generation function utilized by any arbitrary client is immune to cloning or copying. And finally, private keys and the shared secrets of the clients cannot be retrieved from a compromised physical storage. Based on these presumptions, the proposed handshake protocol provides 3 distinct security guarantees.

Guarantee 1 If the handshake protocol is successfully conducted, an attacker cannot obtain the agreed session key. The only way for the attacker to get the session key is to know the private key of at least one of the clients. Throughout the handshake protocol, the session key has never been sent as a message with or without encryption. Potentially, replaying or forging messages does not provide information about IV and encrypted data. Replaying or creating messages 2,3,5 or 6 will result in receiving encrypted messages and data. With this information, it is not possible for an attacker to forge the private key or the session key. Therefore the session key knowledge is secure as long as the private key is not compromised.

Guarantee 2 The protocol guarantees that a unique key is generated in every successful handshake. With our second assumption in mind, clients share a random hexadecimal number of length 16, these values are processed through XOR for a random IV value. Even if the asymmetric key pairs are the same, the IV values are highly unlikely to be the same value for the following session. With different IV values, the same shared secret results in a different encrypted value. A replay attack on the first message or forging a hello message will result in a new random nonce at each attempt. The attacker will get an ecrypted IV value at most which is not enough to know anything about the shared secret.

Guarantee 3 If the client receives the encrypted message with its own nonce, it means that the opposing client can use the public key that they are assigned to therefore achieving authentication. After receiving the message 3, the client decrypts this message with the session key. If that decrypted message is the same as the client nonce that they sent to the opposing client, this means that the client can use the protocol and their corresponding private key to agree on a secret and encrypt the received nonce. An attacker that tries to imitate a client by replaying the message 3 in order to make the other client verify the attacker has to replay/forge the other messages. Replaying the messages does not work since the client nonce will be different (will be the same nonce with negligible probability). Therefore, replaying the message 3 will result in a mismatch between the client nonce and the decrypted message 5. Creating a verifiable message 3 requires the attacker to know the private key of the client that the attacker wants to imitate. Throughout the handshake protocol, the private key or the session key has never been shared with or without encryption. Based on our assumption stating that it is not possible to retrieve the private key or the session key of a client, we claim it is not possible to forge a message that can trick the opposing client into verifying the attacker as any other benign client in the network.

4. Experimental Results

4.1. Avispa-Span Protocol Analysis

To evaluate the security of the proposed handshake protocol against potential security threats, we used the Automated Validation of Internet Security Protocols and Applications (AVISPA) framework, alongside the Security Protocol Animator (SPAN). AVISPA-SPAN performs formal analysis on cryptographic protocols by simulating adversarial behavior and exhaustively checking for common exploits, such as replay attacks, key compromise, and man-in-the-middle intrusions. Our simulation results revealed no identified vulnerabilities within the handshake protocol. The AVISPA-SPAN analysis provides additional confidence in the protocol's ability to maintain secure peer-to-peer communications. The result of this analysis can be seen in Fig. 4.

4.2. Experimental Setup Parameters

To accurately represent real-world IoT network behavior in our experimental environment, we first extracted two essential parameters from a benign IoT PCAP dataset [30]: the average connection count per source-destination IP pair and the average packets per connection. These parameters are critical because they reflect fundamental traffic characteristics, how frequently hosts establish connections and the typical volume of packets exchanged per session. With these parameters we are simulating the behavior of benign IoT devices.

We computed these values by aggregating the PCAP data through group-by operations on the Src IP and Dst IP fields to obtain the number of distinct connections, and by summing forward and backward packets to measure traffic volume. We found that this IoT network traffic has 17.863 packets per connection on average. And average daily connections between two IoT devices are around 55.5993. The resulting averages provide a representation of network load for a benign IoT network traffic.

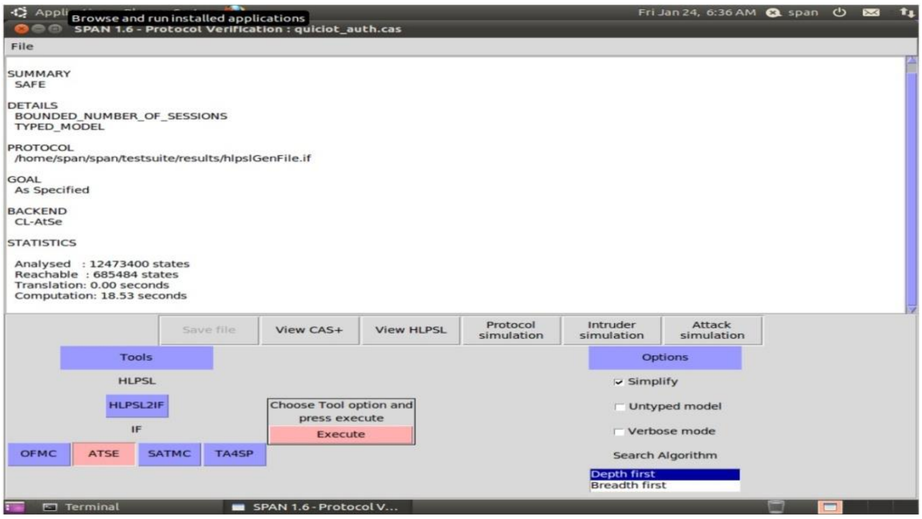
In our NS-3 simulation, we employ these average values to determine both the frequency of messages and the number of connections sustained over a seven-day period, aligning with the typical duration of session ticket validity. By sending the average amount of messages and establishing the average connection count throughout this timeframe, the simulation environment closely replicate observed IoT traffic patterns. This approach ensures our experimental results are grounded in realistic conditions, thus offering a more credible assessment of the proposed protocol's performance in an IoT setting.

In [19] authors simulated the various wireless technologies for their NS-3 simulation using different delay and and network speeds. These wireless network technologies are; wifi, 4G, and satellite networks. They used 12.5, 50, and 300 milliseconds of delay and 20, 10 and 1.5 Mbps speed respectively. These networks all have use in IoT communications. Smart home devices can be an example of wifi, wearable technologies such as smart watches can be an example of 4G-enabled devices, and drones can be an example to the satelliteconnected devices. In our comparison, we use the same setup.

4.3. NS-3 Network Simulation

In this section, we discuss and compare our simulated results against the IoT-PKI [1] authentication protocol simulated in NS-3. Their authentication protocol simulation is implemented using TCP module of NS-3, and the standard TLS 1.2 encryption ciphers(RSA and

AES). We defined 2 IoT Devices and 1 B-node for performance comparison. The simulations for this comparison are conducted using NS-3 which is an open-source discrete-event network simulator for Internet systems. It is capable of simulating and analyzing a wide variety of network setups.



```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/hlpslGenFile.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 12473400 states
  Reachable  : 685484 states
  Translation: 0.00 seconds
  Computation: 18.53 seconds
```

☑ Simplify
□ Untyped model
□ Verbose mode
Search Algorithm: Depth first

Fig.4. Avispa-SPAN Results

In [28] presents a native implementation of the QUIC transport protocol within the NS3 simulation environment. The implementation extends the existing TCP NS-3 codebase to incorporate key QUIC features, including stream multiplexing, low-latency initial handshake, and improved Selective Acknowledgment (SACK) through ACK frames. A unique QUIC socket implementation allows for the integration of both traditional TCP and new QUIC-only congestion control algorithms. The authors validate the performance of their implementation through simulations on a dumb-bell topology, comparing the behavior of legacy congestion control algorithms with a non-legacy, Internet-Draft-based QUIC congestion control. This implementation of QUIC in NS-3 does not cover encryption. We used our proposed handshake protocols over the QUIC protocol for the comparisons.

For our simulations, cryptopp library of C++ have been used. For the DHT, Pyhton Kademlia library has been used and get and set functions have been called inside the NS-3 simulation. NS-3 simulations do not record the computation time taken inside its nodes. Because of this, we have used the Chrono library of C++ to add the time taken within the cryptographic calculations.

In contrast to the work in [1], which employs TLS 1.2 with at best a one round-trip time (1-RTT) session resumption, our protocol leverages QUIC 0-RTT session resumption to minimize handshake overhead. When we integrate our average daily connection establishments and average packet counts per connection to our NS-3 simulation, the 0-RTT connection provides substantial performance gain between 25%-46%. With these simulation parameters, our protocol outperforms the compared work as shown in the Fig. 5.

We used Elliptic curve for asymmetric encryption and Salsa20 for symmetric encryption. Compared to RSA for asymmetric encryption this %72 bandwith reduction when communicating public keys. Also salsa20 provides %42 faster encyption & decryption operation times compared to AES.

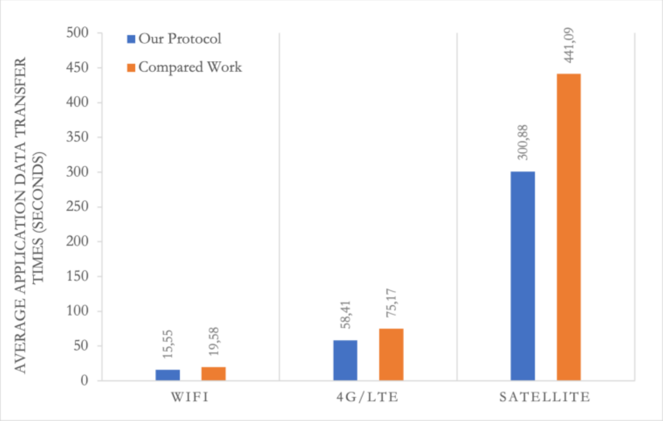


Fig.5. Performance of Authentication Protocols for Various Network Types

5. CONCLUSIONS AND FUTURE WORK

In conclusion, we introduced a decentralized public key infrastructure (DPKI) for IoT devices, leveraging Kademlia-based Distributed Hash Tables (DHT) to prevent single points of failure. Building on this robust authentication layer, our customized QUIC protocol for peer-to-peer IoT communication effectively reduces overhead and latency, thereby catering to the needs of resource-constrained devices. By refining QUIC at the transport layer, we ensure low-latency data transfer, while integrating Elliptic Curve Diffie-Hellman for secure handshakes and Salsa20 for efficient symmetric encryption. This combination of distributed authentication and lightweight cryptographic mechanisms delivers both high performance and strong security for IoT environments.

Looking ahead, our future work will focus on further securing this decentralized authentication framework using blockchain technologies. We also plan to introduce additional features, such as authorization and secure updating or deletion of DHT entries, addressing NAT traversal barriers and dynamic DHT churn, potentially leveraging zero-knowledge proofs in edge computing systems.

REFERENCES

[1] J. Won, A. Singla, E. Bertino, and G. Bollella, “Decentralized Public Key Infrastructure for Internetof-Things,” in *MILCOM 2018 - IEEE Military Communications Conf. (MILCOM)*, Los Angeles, CA, 2018, pp. 907–913, doi: 10.1109/MILCOM.2018.8599710.

[2] D. Pavithran and K. Shaalan, “Towards Creating Public Key Authentication for IoT Blockchain,”in *6th HCT Information Technology Trends (ITT)*, Ras Al Khaimah, UAE, 2019, pp. 110–114, doi: 10.1109/ITT48889.2019.9075105.

[3] A. Yakubov, W. M. Shbair, A. Wallbom, D. Sanda, and R. State, “A blockchain-based PKI managementframework,” in *NOMS 2018 - IEEE/IFIP Network Operations and Management Symp.*, Taipei, Taiwan, 2018, pp. 1–6, doi: 10.1109/NOMS.2018.8406325.

[4] E. Kfoury and D. Khoury, “Securing NATted IoT Devices Using Ethereum Blockchain and DistributedTURN Servers,” in *10th Int. Conf. Advanced Infocomm Technology (ICAIT)*, Stockholm, Sweden, 2018, pp. 115–121, doi: 10.1109/ICAIT.2018.8686623.

[5] A. Singla and E. Bertino, “Blockchain-Based PKI Solutions for IoT,” in *4th IEEE Int. Conf. Collaboration and Internet Computing (CIC)*, Philadelphia, PA, 2018, pp. 9–15, doi: 10.1109/CIC.2018.00-45.

[6] A. Pino, D. Margaria, and A. Vesco, “Combining Decentralized IDentifiers with Proof of Membershipto Enable Trust in IoT Networks,” in *33rd Int. Telecommunication Networks and*

Applications Conf., Melbourne, Australia, 2023, pp. 310–317, doi: 10.1109/ITNAC59571.2023.10368540.

[7] D. G. Berbecaru and L. Pintaldi, "Exploiting Emercoin Blockchain and Trusted Computing for IoTScenarios: A Practical Approach," in *IEEE Symp. on Computers and Communications (ISCC)*, Gammarth, Tunisia, 2023, pp. 771–776.

[8] M. Fan, Z. Zhang, Z. Li, G. Sun, H. Yu, and M. Guizani, "Blockchain-Based Decentralized andLightweight Anonymous Authentication for Federated Learning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 12075–12086, Sept. 2023, doi: 10.1109/ISCC58397.2023.10217961.

[9] N. Cardamone *et al.*, "Blockchain-Based Public Key Authentication of IoT Devices for Electrical Energy Systems," in *AEIT Int. Annual Conf. (AEIT)*, Rome, Italy, 2022, pp. 1–6, doi: 10.23919/AEIT56783.2022.9951818.

[10] S. Hameed *et al.*, "A Scalable Key and Trust Management Solution for IoT Sensors Using SDN and Blockchain Technology," *IEEE Sensors J.*, vol. 21, no. 6, pp. 8716–8733, Mar. 2021, doi: 10.1109/JSEN.2021.3052009.

[11] S. N, V. Bhat K, and M. Rajarajan, "Blockchain-based Scheme for Authentication and Capabilitybased Access Control in IoT Environment," in *11th IEEE Ann. Ubiquitous Computing, Electronics & Mobile Comm. Conf. (UEMCON)*, New York, NY, 2020, pp. 323–330, doi: 10.1109/UEMCON51285.2020.9298116.

[12] Z. A. Lux, D. Thatmann, S. Zickau, and F. Beierle, "Distributed-Ledger-based Authenticationwith Decentralized Identifiers and Verifiable Credentials," in *2nd Conf. Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*, Paris, France, 2020, pp. 71–78, doi: 10.48550/arXiv.2006.04754

[13] L. D. Santis, V. Paciello, and A. Pietrosanto, "Blockchain-Based Infrastructure to Enable Trust in IoTEnvironment," in *IEEE Int. Instrumentation and Measurement Technology Conf. (I2MTC)*, Dubrovnik, Croatia, 2020, pp. 1–6, doi: 10.1109/I2MTC43012.2020.9128817.

[14] Y. Chu, J. M. Kim, Y. Lee, S. Shim, and J. Huh, "SS-DPKI: Self-Signed Certificate Based Decentralized Public Key Infrastructure for Secure Communication," in *IEEE Int. Conf. Consumer Electronics (ICCE)*, Las Vegas, NV, 2020, pp. 1–6, doi: 10.1109/ICCE46568.2020.9043086.

[15] L. Perugini and A. Vesco, "On the Integration of Self-Sovereign Identity with TLS 1.3 Handshake toBuild Trust in IoT Systems," *Internet of Things*, vol. 25, 2024, 101103, doi:10.1016/j.iot.2024.101103.

[16] Y. Zheng, W. Liu, C. Gu, and C.-H. Chang, "PUF-Based Mutual Authentication and Key ExchangeProtocol for Peer-to-Peer IoT Applications," *IEEE Trans. Dependable and Secure Comput.*, vol. 20, no. 4, pp. 3299–3316, Jul./Aug. 2023, doi: 10.1109/TDSC.2022.3193570.

[17] P. Flood and M. Schukat, "Peer to peer authentication for small embedded systems: A zero-knowledgebased approach to security for the Internet of Things," in *10th Int. Conf. Digital Technologies*, Zilina, Slovakia, 2014, pp. 68–72, doi: 10.1109/DT.2014.6868693.

[18] F. Iqbal, M. Gohar, H. Karamti, W. Karamti, S.-J. Koh, and J.-G. Choi, "Use of QUIC for AMQP inIoT Networks," *Computer Networks*, vol. 225, 2023, 109640, doi: 10.1016/j.comnet.2023.109640,

[19] F. Fern´andez, M. Zverev, P. Garrido, J. R. Jua´rez, J. Bilbao, and R. Agu¨ero, "And QUICmeets IoT: Performance Assessment of MQTT over QUIC," in *16th Int. Conf. Wireless and Mobile Computing, Networking and Communications (WiMob)*, Thessaloniki, Greece, 2020, pp. 1–6, doi: 10.1109/WiMob50308.2020.9253384.

[20] J.-H. Jung, H.-B. Nam, D.-K. Choi, S.-J. Koh, "Use of QUIC for CoAP Transport in IoT Networks," *Internet of Things*, vol. 24, 2023, 100905, doi: 10.1016/j.iot.2023.100905.

[21] A. Pagani, "QUIC Bitcoin: Fast and Secure Peer-to-Peer Payments and Payment Channels," in*IEEE Future Networks World Forum (FNWF)*, Montreal, QC, Canada, 2022, pp. 578–584, doi: 10.1109/FNWF55208.2022.00107

[22] C. Fromknecht, D. Velicanu, and S. Yakoubov, "Certcoin: A Namecoin Based Decentralized Authentication System," MIT Tech. Rep., 2014, pp. 1–10.

[23] S. Kumari, *et al.*, "A Secure Authentication Scheme Based on Elliptic Curve Cryptography for IoT and Cloud Servers," *J. Supercomput.*, vol. 74, pp. 6428–6453, 2018, doi: 10.1007/s11227-017-2048-0

[24] A. Tewari and B. B. Gupta, "A Lightweight Mutual Authentication Protocol Based on Elliptic CurveCryptography for IoT Devices," *Int. J. Adv. Intell. Paradigms*, vol. 9, no. 2–3, pp. 111–121, 2017, doi: 10.1109/CONFLUENCE.2018.8442962.

[25] E. Lara, *et al.*, "A Lightweight Cipher Based on Salsa20 for Resource-Constrained IoT Devices," *Sensors*, vol. 18, no. 10, 2018, Art. no. 3326, doi: https://doi.org/10.3390/s18103326

[26] B. Dowling, *et al.*, “A Cryptographic Analysis of the TLS 1.3 Handshake Protocol,” *J. Cryptol.*, vol. 34, no. 4, 2021, Art. no. 37, doi: 10.1145/2810103.2813653.

[27] Y. Yu, M. Xu, and Y. Yang, “When QUIC Meets TCP: An Experimental Study,” in *36th Int. Performance Computing and Communications Conf. (IPCCC)*, San Diego, CA, 2017, pp. 1–8, doi: 10.1109/PCCC.2017.8280429.

[28] A. De Biasio, F. Chiariotti, M. Polese, A. Zanella, M. Zorzi, ”A QUIC Implementation for ns-3”,Proceedings of the Workshop on ns-3 (WNS3 ’19), Firenze, Italy, 2019, doi: 10.1145/3321349.3321351.

[29] D. Dolev and A. Yao, “On the Security of Public Key Protocols,” *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983, doi: 10.1109/TIT.1983.1056650.

[30] *Dataset of Legitimate IoT Data*, data.gouv.fr. [Online]. Available: https://www.data.gouv.fr/en/datasets/dataset-of-legitimate-iot-data/ [Accessed: Dec. 20, 2024].

[31] A. Yakubov, W. M. Shbair, A. Wallbom, D. Sanda and R. State, ”A blockchain-based PKI managementframework,” NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 2018, pp. 1-6, doi: 10.1109/NOMS.2018.8406325.

[32] W. Trappe, R. Howard and R. S. Moore, ”Low-Energy Security: Limits and Opportunities in theInternet of Things,” in IEEE Security & Privacy, vol. 13, no. 1, pp. 14-21, Jan.-Feb. 2015, doi: 10.1109/MSP.2015.7.

[33] Faheem Iqbal, Moneeb Gohar, Hani Alquhayz, Seok-Joo Koh, Jin-Ghoo Choi, Performance evaluationof AMQP over QUIC in the internet-of-thing networks, Journal of King Saud University - Computer and Information Sciences, Volume 35, Issue 4, doi: 10.1016/j.jksuci.2023.02.018.

AUTHORS

Ozan Tarlan Ozan Tarlan received both the B.Sc. and M.Sc. degrees in Computer Science from Ozye˘gin University, Istanbul, Tu¨rkiye. He is currently pursuing a Ph.D. at the¨ same institution. His research interests include blockchain technologies, network security, and decentralized systems.

Kubra Kalkan Cakmakcı Dr. Kubra Kalkan received her BSc and MSc degrees in Computer Science and Engineering department from Sabanci University in 2009 and 2011, respectively. She received her PhD degree from Bogazici University in 2016. After PhD, she worked as a post doctoral researcher in University of Oxford. She also worked as a visiting researcher at various institutions such as Ecole Polytechnique F´ed´erale de Lausanne´ (EPFL), Microsoft Redmond and Northeastern University during her MSc and PhD. Her PhD thesis is rewarded IEEE Turkey PhD Thesis Award and Bogazici University BAP PhD Thesis Award. Her current research interests include network security, computer networks, wireless networks, IoT, P2P networks and software-defined networking (SDN).

Ilgin Safak Ilgin Safak received her B.S. degree from Baskent University, Ankara, Turkey, in 2002 in Electrical and Electronics Engineering; M.S. and Ph.D. degrees from Hacettepe University, Ankara, Turkey, in Electrical and Electronics Engineering in 2006 and 2013, respectively. Her research interests include coding theory, information theory and statistical signal processing in wireless communications systems.

Hasan Sozer Hasan S¨ozer received his B.Sc. and M.Sc. degrees in computer engineering from Bilkent University, Turkey, in 2002 and 2004, respectively. He received his Ph.D. degree in 2009 from the University of Twente, The Netherlands. From 2002 until 2005, he worked as a software engineer at Aselsan Inc. in Turkey. From 2009 until 2011, he worked as a post-doctoral researcher at the University of Twente. He has been working as a faculty member at Ozyegin University since 2011.