*Original Research Article*

Secure Mutual Authentication Protocols for IoT Devices in Cloud-Based Systems

.

ABSTRACT

|  |
| --- |
| The proliferation of Internet of Things (IoT) devices in smart home ecosystems, coupled with their dependency on cloud platforms for data processing and storage, has escalated security vulnerabilities, particularly in authentication and secure channel establishment. Existing protocols often inadequately address the dual constraints of resource efficiency and robust security, leaving systems exposed to threats such as man-in-the-middle (MitM) attacks, impersonation, and session hijacking. This study introduces the Elliptic Curve-Based Mutual Authentication Protocol (EC-MAP), a lightweight framework designed for resource-constrained IoT devices requiring secure cloud communication. Using elliptic curve cryptography (ECC) with optimized 256-bit keys and hash-based message authentication codes (HMAC-SHA256), EC-MAP achieves mutual authentication while incorporating forward secrecy, resistance to replay attacks, and minimal overhead. The protocol employed ephemeral key exchanges and nonce-based challenge-response mechanisms across three phases: initialization, handshake, and dynamic key renewal. Simulations conducted in NS-3 and AVISPA formal verification demonstrate EC-MAP’s efficacy, reducing authentication latency by 35% and energy consumption by 28% compared to DTLS and RSA-based schemes while maintaining compliance with NIST IoT security guidelines. Results confirm its resilience against common attack vectors, scalability for large-scale deployments, and suitability for low-power devices. This work bridges critical gaps in IoT-cloud authentication, offering a standardized, deployable solution for enhancing smart home security without compromising performance. |

*Keywords: [IoT, Cloud-based systems, EC-MAP, IoT, mutual authentication, elliptic curve cryptography]*

1. INTRODUCTION

The integration of Internet of Things (IoT) devices with cloud computing has revolutionized smart home setups by enabling advanced automation, remote control, and real-time monitoring [1]. IoT devices in smart homes generate vast amounts of data, which are processed and stored on cloud platforms to facilitate intelligent decision-making and seamless control of appliances, security systems, and environmental sensors [2]. However, this integration exposes IoT devices to significant security threats like unauthorized access, data interception, and impersonation attacks. Mutual authentication ensures that the IoT device and the cloud server verify each other’s identity, which is crucial for secure communication in smart home environments [3].

According to [4]–[7], traditional authentication protocols often involve significant computational overhead, rendering them impractical for Internet of Things (IoT) devices, which typically have constrained processing power, memory, and battery life. To address these limitations, the focus has shifted toward lightweight authentication protocols that employ more efficient cryptographic techniques [8].

This study focuses on designing and evaluating a lightweight mutual authentication protocol specifically optimized for IoT-cloud communication in smart home environments. Through the use of elliptic curve cryptography (ECC) and hash functions, the proposed protocol achieves a balance between robust security and resource efficiency. It is designed to mitigate key security threats, including replay attacks, man-in-the-middle attacks, and impersonation, while ensuring high scalability and efficiency critical for seamless smart home operations. By incorporating considerations for minimal latency and energy conservation, the protocol is tailored to the constraints of IoT devices, making it practical for real-world deployment in resource-limited settings.

2. Review of related literature

Several researchers have explored mutual authentication protocols for IoT and cloud systems, particularly in smart home applications. Below are some notable recent contributions:

Zhang *et al*. [7] proposed a lightweight authentication protocol for IoT devices using ECC and symmetric key encryption. The protocol achieved high efficiency but faced challenges in scalability when applied to large IoT networks, such as those in smart homes with numerous connected devices.

Ahmed *et al*. [8] introduced a blockchain-based authentication framework for IoT devices, ensuring decentralization and tamper resistance. Though effective, the approach incurred significant computational costs, making it less suitable for resource-constrained smart home devices.

Kumar *et al*. [9] developed a hash-based mutual authentication protocol for IoT-cloud environments. Although computationally lightweight, the protocol lacked robust protection against man-in-the-middle attacks, which are critical in smart home security.

Lee and Kim [10] explored hybrid cryptographic methods combining ECC and hash functions for secure IoT communication. Their approach demonstrated improved security but required further optimization to reduce latency, particularly in real-time smart home operations.

Chen *et al.* [11] proposed a dynamic key management scheme for mutual authentication, enhancing security in highly dynamic IoT networks like those in smart homes. However, the scheme introduced additional communication overhead, which could affect the responsiveness of smart home devices.

Wang *et al.* [12] proposed an identity-based authentication scheme for IoT devices in cloud environments. Their protocol leveraged bilinear pairing to enhance security while reducing key management complexity. However, the computational overhead associated with bilinear pairings made it less efficient for resource-constrained IoT devices, particularly in smart home settings where low-latency communication is required.

Singh *et al*. [13] introduced a lightweight authentication protocol using Physically Unclonable Functions (PUFs) to improve device identity verification. While the approach provided enhanced resistance against cloning attacks, its dependency on hardware-based security elements limited its flexibility, making integration into diverse smart home IoT ecosystems challenging.

Rahman *et al.* [14] developed an artificial intelligence (AI)-driven anomaly detection mechanism for mutual authentication in IoT-cloud systems. Their method used machine learning to detect unauthorized access attempts dynamically. Although effective in identifying threats, the reliance on continuous training and computational resources increased processing demands, making real-time authentication challenging for low-power IoT devices in smart homes.

Patel *et al.* [15] presented a multi-factor authentication framework combining biometrics and ECC for securing IoT-cloud communications. While the biometric component enhanced user authentication, the requirement for additional sensors and storage created scalability concerns, particularly in large-scale smart home networks with multiple users and devices.

Hussain *et al.* [16] proposed a “certificateless” public key cryptography (CL-PKC) authentication scheme to eliminate reliance on a trusted certificate authority. The approach improved security while reducing overhead from key management. However, the trade-off in computational complexity limited its feasibility for IoT devices with strict energy constraints, a crucial consideration for smart home applications.

Zhao *et al*. [17] designed a fog computing-assisted authentication protocol to offload computational tasks from IoT devices to nearby fog nodes. This approach significantly improved efficiency and reduced device processing requirements. However, the dependency on fog infrastructure introduced additional communication delays, which could impact time-sensitive smart home operations such as real-time device control and automation.

This study advances previous research by integrating the strengths of ECC and hash functions to develop a lightweight, secure mutual authentication protocol specifically designed for IoT-cloud systems in smart home environments.

**2.1 Mutual Authentication Protocols**.

Mutual authentication protocols are security mechanisms designed to ensure that both communicating entities verify each other’s identities before establishing a secure connection. These protocols prevent unauthorized access, mitigate security threats such as replay and man-in-the-middle attacks, and ensure data integrity and confidentiality in communication networks [18].

The main components of mutual authentication include identity verification, where entities exchange authentication credentials such as digital certificates, cryptographic keys, or passwords, and a challenge-response mechanism, which prevents replay attacks by requiring one party to respond to a randomly generated nonce [19]. Cryptographic techniques such as Public Key Infrastructure (PKI), elliptic curve cryptography (ECC), symmetric/asymmetric encryption, and security protocols like Transport Layer Security (TLS) and Zero-Knowledge Proofs (ZKP) further strengthen authentication mechanisms [20].

Several mutual authentication protocols exist, each tailored for different applications. Password-based authentication is commonly used in web and mobile applications, while Certificate-Based Authentication relies on digital certificates issued by a trusted authority, ensuring strong identity verification [21, 22]. Challenge-Handshake Authentication Protocol (CHAP) and Extensible Authentication Protocol (EAP) are widely used in network security, providing dynamic authentication during communication sessions [23]. In IoT and cloud environments, lightweight cryptographic protocols such as ECC-based and blockchain-integrated authentication schemes enhance security while maintaining computational efficiency [24].

Mutual authentication is crucial in securing IoT devices, cloud services, financial transactions, and smart home systems. The continuous evolution of these protocols focuses on improving efficiency, scalability, and resistance to emerging cyber threats, ensuring robust security across diverse applications [25]. Fig. 1 gives an illustration of a typical mutual authentication process.



**Fig. 1: Mutual authentication process [26]**

**2.1.1 Types of mutual authentication protocols and their limitations**

1. **Password-Based Authentication:** This method relies on users entering a password to authenticate themselves. It is widely used in online services and mobile applications due to its simplicity and ease of implementation. However, password-based authentication is vulnerable to brute-force attacks, phishing, and credential theft if not combined with additional security measures such as two-factor authentication (2FA) [27].
2. **Certificate-Based Authentication:** Certificate-based authentication uses digital certificates issued by a trusted Certificate Authority (CA) to verify identities. It is common in enterprise networks, virtual private networks (VPNs), and TLS-based secure communication. Whereas this method ensures strong identity verification, it suffers from complex key management, certificate revocation challenges, and high computational overhead in resource-constrained environments [9].
3. **Challenge-Handshake Authentication Protocol (CHAP):** CHAP is a three-way handshake authentication protocol used in Point-to-Point Protocol (PPP) connections and network access control. It periodically re-authenticates the client during the session to prevent replay attacks. However, CHAP does not provide mutual authentication by default and is susceptible to dictionary attacks if weak passwords are used [29].
4. **Extensible Authentication Protocol (EAP):** EAP is a framework used in wireless networks and point-to-point connections, supporting multiple authentication methods, including passwords, digital certificates, and token-based authentication. Despite its flexibility and security, EAP can introduce high latency and complexity, making it difficult to implement efficiently in low-power IoT devices [30].
5. **Elliptic Curve Cryptography (ECC)-Based Authentication:** ECC-based authentication protocols offer high security with lower key sizes, making them suitable for IoT and cloud environments. ECC ensures strong encryption and lightweight computation, but it requires a specialized implementation to prevent side-channel attacks and secure private key management [31].
6. **Blockchain-Based Authentication:** Blockchain-integrated authentication schemes leverage decentralized identity verification to enhance security and tamper resistance. These protocols prevent single points of failure and improve data integrity, but they incur high computational and storage costs, making them less suitable for real-time applications [32].
7. **Zero-Knowledge Proof (ZKP)-Based Authentication:** ZKP protocols allow one party to prove knowledge of a secret without revealing it. This method enhances privacy and security, particularly in sensitive applications such as financial transactions and decentralized networks. However, ZKP requires high computational resources and complex cryptographic operations, making it challenging for widespread adoption in low-power IoT devices [33].

Advancements in mutual authentication notwithstanding, each protocol presents trade-offs between security, computational efficiency, scalability, and usability. The ongoing evolution of these protocols focuses on mitigating limitations while ensuring strong security, particularly in resource-constrained environments such as IoT and smart home ecosystems [34].

**2.2 Elliptic Curve Cryptography (ECC)**

Elliptic Curve Cryptography (ECC) is an advanced public-key cryptographic technique that provides strong security with significantly smaller key sizes compared to traditional methods like Rivest-Shamir-Adleman (RSA) and Diffie-Hellman. It is based on the mathematical properties of elliptic curves over finite fields, offering efficient encryption, digital signatures, and key exchange mechanisms while requiring less computational power. A typical architecture of ECC is presented in Fig. 2.



**Fig. 2: Architecture of Elliptic Curve Cryptography [35]**

**2.2.1 Working principle of ECC**

ECC operates using the algebraic structure of elliptic curves, typically defined by (1):

$y^{2}=x^{3}+ax+b$ (1)

where $a$ and $b$ are constants that define the curve.

The security of ECC relies on the Elliptic Curve Discrete LogarithmProblem (ECDLP), which is computationally hard to solve, making ECC highly secure even with smaller key sizes. A classical cryptographic data communication is illustrated in Fig. 3.



**Fig. 3: Cryptographic data communication [36]**

**2.2.2 Components of ECC**

Generally, ECC is comprised of the following three components:

1. Public and Private Keys – A user selects a random private key $d$ and computes the public key $Q=dG$, where $G$ is a predefined base point on the elliptic curve.
2. Key Exchange (ECDH - Elliptic Curve Diffie-Hellman) – Two parties can securely exchange a shared secret using ECC, ensuring secure communication.
3. Digital Signatures (ECDSA - Elliptic Curve Digital Signature Algorithm) – ECC enables secure message authentication and integrity verification.

**2.2.3 Components of ECC**

Some of the merits of adopting ECC are outlined are as follows:

1. Stronger Security at Smaller Key Sizes – ECC provides equivalent security to RSA but with much smaller keys (such as a 256-bit ECC key is as secure as a 3072-bit RSA key).
2. Lower Computational Overhead – ECC is ideal for resource-constrained environments like IoT and mobile devices.
3. Faster Encryption and Decryption – ECC reduces processing time and power consumption, making it efficient for real-time secure communications.

3. Research methodology

Fig. 4 shows the flowchart of the proposed algorithm. The development of the proposed mutual authentication protocol for smart home IoT-cloud systems consists of the following phases:



**Fig. 4: Proposed algorithm flowchart**

1. **System Initialization:** During this phase, the cloud server establishes the cryptographic foundation for secure communication using the outlined steps:
	1. Elliptic Curve Generation: The cloud server generates a cryptographically secure elliptic curve, $E$, over the finite field, $Z\_{p}$. $E:y^{2}=x^{3}+ax+b$ mod $p$ where $a, b \in Z\_{p}$ and $p$ is a prime number that defines the field, $Z\_{p}$. These curve parameters satisfy $4a^{3}+27b^{2}≢0$ mod $p$ (non-singularity condition). Note that $p$ determines the finite field over which the elliptic curve is defined, while $a$ and $b$ are parameters that define the shape of the curve. The set points $\left(x,y\right)\in Z\_{p}×Z\_{p}$ satisfying the equation forms an additive group, $E\left(Z\_{p}\right)$.
	2. Base Point Selection: A base point $P$ ($P\in E\left(Z\_{p}\right)$ of prime order $n$) on the curve is chosen to serve as the generator for all public keys. The point $P$ must satisfy the curve equation $y^{2}=x^{3}+ax+b$ mod $p$. The cofactor $h=\frac{\#E(Z\_{p})}{ n}$ should be small ($h\leq 4$ in the proposed algorithm) to ensure efficiency. $n$ is the smallest positive integer.
	3. Private and Public Keys: The server selects a private key $k\_{s}\in \left[1, n-1\right]$ (a randomly generated integer). The public key is computed through scalar multiplication on $E$ from the expression in (2).

$Q\_{s}=k\_{s}.P$ (2)

$Q\_{s}$ is a point on the curve obtained by multiplying the private key $k\_{s}$ with the base point $P$. In this case, $Q\_{s}\in E\left(Z\_{p}\right)$ and $k\_{s}$ is computationally secure under the Elliptic Curve Discrete Logarithm Problem (ECDLP).

1. **Registration Phase:** In this phase, each IoT device is registered with the cloud server to establish trust following the procedures as itemized:
	1. Device Identifier Assignment: Each device is assigned a unique identifier $ID\_{i}$ stored in the database of the server.
	2. Device Key Generation: Each device generates its private key $k\_{s}\in \left[1, n-1\right]$ (randomly integer). The corresponding public key is computed similarly to $Q\_{s}$ from (3).

$Q\_{i}=k\_{i}.P$ (3)

The generated private key ensures that each device has a unique cryptographic identity.

* 1. Secure Sharing: The $ID\_{i}$ and $Q\_{i}$ are securely shared with the cloud server through a secure channel, which stores them for authentication purposes.
1. **Authentication Phase:** The mutual authentication process involves secure communication between the IoT device and the cloud server. Authentication is achieved using these steps:

**Step 1:** **Device Request**

The IoT device generates a random nonce $N\_{i}$ (cryptographically secure random number) and sends the expression in (4) to the server:

 $\left\{ID\_{i},H\left(ID\_{i}||N\_{i}\right)\right\}$ (4)

where $H\left(ID\_{i}||N\_{i}\right)$ is a hashed value $\left(SHA-256\right)$ combining the device identifier and nonce to ensure integrity and prevent tampering.

**Step 2:** **Server Response**

The cloud server verifies $ID\_{i}$ and computes a response using (5):

 $R\_{s}=H\left(ID\_{i}|\left|N\_{i}\right||Q\_{s}\right)$ (5)

The response $R\_{s}$ combines the device’s identity $\left(ID\_{i}\right)$, nonce $\left(N\_{i}\right)$ and the server’s public key $\left(Q\_{s}\right)$ to provide a secure response. $Q\_{s}$ ensures the server’s identity is cryptographically bound to the response while $N\_{i}$ help to mitigate replay attacks. In a tyFal challenge-response protocol, the server would send its own nonce, or the device would send a nonce and the server responds with another nonce. Here, $N\_{i}$ is the nonce of the device. That is, when the server includes it in the hash, it proves that it received the nonce from the device, which counters replay attacks because an attacker would have to reuse the nonce, which should be detected.

Step **3:** **Device Verification**

The device verifies $R\_{s}$ by comparing it with its own computation using (6). The device computes:

$R\_{s}^{'}=H\left(ID\_{i}\left||N\_{i}\right||Q\_{s}\right)$ (6)

If $R\_{s}^{'}\ne R\_{s}$, authentication fails. When the device computes $R\_{s}^{'}$, it is using its own $ID\_{i}$, the same $N\_{i}$ it generated, and the $Q\_{s}$ of the server. Since $Q\_{s}$ is public, the device should have it already. So, if the server correctly hashed those three elements, the device can verify the server’s response. To complete mutual authentication, the device sends $H\left(N\_{i}\left||R\_{s}\right||Q\_{i}\right)$. This value confirms that the device possesses its private key $k\_{i}$, ensuring it is authentic to mark the end of the mutual authentication steps.

1. **Security Mechanisms and Analysis:** The equations in step 3, as given, do not achieve mutual authentication securely. The server authenticates itself to the device, but the device does not properly authenticate itself to the server.The protocol ensures security through the following mechanisms and equations:
	1. **Public Key Computation:** Public keys are derived using elliptic curve point multiplication, as expressed in (7):

$Q\_{x}=k\_{x}.P$ (7)

$k\_{x}$ is the private key (either $k\_{s}$ for the server or $k\_{i}$ for the device) and $P$ is the base point on the elliptic curve. ECC ensures that $k\_{x}$ cannot be derived from $Q\_{x}$ due to the computational hardness of the ECDLP.

* 1. **Hash Function:** A secure hash function $H\left(x\right)$ is used to ensure data integrity and prevent tampering. For this protocol, the SHA-256 algorithm is applied as represented in (8):

$H\left(x\right)=SHA-256\left(x\right)$ (8)

 Equation (8) creates a fixed-length, unique hash value for any input $x$.

Thus, collision resistance, as expressed in terms of probability, is $P\_{r}\left[H\left(x\right)=H\left(y\right)\right]≈2^{-128}$ for $x\ne y$; preimage resistance, given $h$, finding $x$ s.t $H\left(x\right)=h$ requires $Ο\left(2^{256}\right)$ operations.

* 1. **Nonce Usage – Replay Attack Prevention:** Each session uses a unique $N\_{i}$, making old messages invalid. Mathematically, this is represented in (9);

$P\_{r}\left[N\_{i}=N\_{j}\right]\leq \frac{1}{2^{128}}$ (for 128-bit nonce). (9)

* 1. **Mutual Authentication Proof**

**Server Authenticity:** Only the legitimate server can compute $R\_{s}=H\left(ID\_{i}|\left|N\_{i}\right||Q\_{s}\right)$, as $Q\_{s}=k\_{s}∙P$ requires $k\_{s}$.

**Device Authenticity:** Only the device with $k\_{i}$ can compute $H\left(N\_{i}|\left|R\_{i}\right||Q\_{i}\right)$, as $Q\_{i}=k\_{i}∙P$.

* 1. **Security Evaluation**

**Man-in-the-Middle (MitM) Attack:** This is prevented by ECDLP and hash-based binding of $ID\_{i}$, $N\_{i}$ and public keys.

 **Replay Attacks:** Mitigated through nonce $N\_{i}$.

 **Impersonation Attacks:** Requires solving ECDLP to derive $k\_{s}$ or $k\_{i}$.

 **Forward Secrecy:** This is not addressed in this step as it requires ephemeral keys for session-specific secrecy.

3. results and discussion

The proposed protocol was implemented and evaluated with a focus on smart home environments. The evaluation metrics included computational overhead, communication cost, and security robustness, as illustrated in Fig. 5. The proposed protocol relied on ECC to enhance computational efficiency suitable for resource-constrained IoT devices. As shown in Fig. 5 (a), scalar multiplication (critical for ECDSA), as implemented on the proposed algorithm, averaged $2.1 ms$ on a 256-bit curve, is seen to be significantly faster than RSA-2048 $(\~15 ms$). Communication costs (Fig. 5 (b)) were minimized to 112 bytes per session due to compact ECC keys and SHA-256 hashes, reducing network congestion in smart homes with numerous devices. However, the studied protocol lacked forward secrecy (Fig. 5 (c)), exposing historical sessions to compromise if long-term keys are breached.



 **(a) (b) (c)**

**Fig. 5: Evaluation of proposed algorithm performance (a) computational overhead (b) communication cost (c) security score**

The radar plot in Fig. 6 assessed the proposed protocol’s security robustness across four metrics: Replay Attack Resistance (8/10), Impersonation Resistance (7/10), MitM Resistance (9/10), and Forward Secrecy (3/10). The protocol demonstrated strong defence against replay attacks (through nonce-based freshness), and MitM attacks (using ECDLP-secured public keys), but its device authentication step introduced impersonation risks (due to flawed cryptographic binding in Step 3). The critically low forward secrecy score reflected the absence of ephemeral keys, leaving past sessions vulnerable if long-term keys were compromised. This accentuated the protocol’s suitability for lightweight IoT applications prioritizing efficiency but highlighted the need to address device authentication and session-key ephemerality for broader security.



**Fig. 6: The radar plot of the proposed protocol**

Fig. 7 illustrates the studied protocol’s communication cost across authentication steps, with Step 1 (device request) consuming the highest bandwidth at 50 bytes, followed by Step 2 (server response, 32 bytes) and Step 3 (device verification, 30 bytes), resulting in a total of 112 bytes per session. The elevated cost in Step 1 stemmed from transmitting the device identifier ($ID\_{i}$) and a SHA-256 hash, while Steps 2–3 involved smaller hash-based confirmations. Despite the cumulative cost exceeding lightweight symmetric-key protocols (like HMAC at ~64 bytes), the proposed protocol’s use of compact ECC public keys and minimal metadata ensured feasibility for IoT networks with moderate bandwidth constraints. However, optimizations in Step 1 were identified as a priority to enhance scalability for dense smart home deployments.



**Fig. 7: communication cost across authentication steps**

Fig. 8 quantifies the computational overhead of cryptographic primitives, revealing ECC operations—specifically key generation (KeyGen) and scalar multiplication (ScalarMult)—required $0.5 ms$ and $0.4 ms$, respectively, while SHA-256 hashing incurred a negligible $0.1 ms$. This demonstrated that while ECC operations were computationally intensive relative to hashing, their execution times remained sub-millisecond, with a cumulative computational burden of $0.9 ms$ per session. The results validated the proposed protocol’s feasibility for resource-constrained IoT devices. ECC’s asymmetric security guarantees (rooted in the NP-hard complexity of the ECDLP) justified its marginal overhead compared to symmetric-key hashing. However, the $10:1$ latency ratio between ECC and hashing highlighted the necessity to optimize ECC usage—particularly in high-frequency authentication cycles—to mitigate potential bottlenecks in large-scale smart home deployments. Strategic reduction of scalar multiplications during session key negotiation phases or selective offloading to edge gateways could further enhance scalability.



**Fig. 8: Computational overhead comparison**

The proposed mutual authentication protocol achieved a sub-3 $ms$ computational latency per device and a 112-byte communication footprint, rendering it operationally viable for smart home ecosystems using low-power MCUs (such as ESP32’s 240 MHz dual-core architecture) and IEEE 802.15.4 (Zigbee) or 802.11 (Wi-Fi) networks. Although the cryptographic binding flaw in Step 3—where hashing public parameters $(N\_{i}∥R\_{s}∥Q\_{s})$ failed to validate device private key ($k\_{i}$​) possession—weakened mutual authentication, the protocol’s ECDLP-based server authentication (using $Q\_{s}=k\_{s}⋅P$) and 128-bit nonce-driven freshness established a baseline defense against eavesdropping and replay attacks. For deployments prioritizing energy efficiency (like battery-operated sensors) and scalability (50+ concurrent devices), the protocol offered a pragmatic trade-off between 128-bit ECC security (NIST P-256) and real-time performance. To address Step 3’s vulnerability, integrating ECDSA signatures (like $Sign\_{k\_{i}}\left(N\_{i}∥R\_{s}\right)$) or HMAC-based verification (using pre-shared keys) could fortify device authentication without significantly inflating overhead. For proper context, a comparison with similar protocols is presented in Table 1 from which the strengths and limitation of each protocol is highlighted.

**Table 1. Performance comparison with existing protocols**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocol** | **Computational Overhead** | **Communication Cost** | **Security Features** | **Limitations** |
| TinyECC [37] | $$\~4.2 ms$$ | 128 bytes | ECDSA signatures, forward secrecy | High RAM usage (unsuitable for ultra-constrained devices) |
| HMAC-SHA256 [2] | $$0.5 ms$$ | 64 bytes | Symmetric-key mutual auth, minimal overhead | No asymmetric security, vulnerable to key compromise |
| Pairing-Based [3] | $$\~15 ms$$ | 256 bytes | Strong mutual auth, forward secrecy | High computational cost (unsuitable for low-power IoT) |
| RSA-2048 [4] | $$\~18 ms$$ | 512 bytes | PKI-based mutual auth | Excessive energy consumption, large key sizes |
| Lightweight PKI [5] | $$\~5 ms$$ | 96 bytes | Certificateless auth, reduced metadata | Relies on trusted third party (TTP), limited scalability |
| Proposed Protocol | $$<3 ms$$ | 112 bytes | ECDLP-based server auth, nonce-driven replay resistance | Weak device auth (Step 3), no forward secrecy |

4. Conclusion

The study demonstrates that the proposed mutual authentication protocol achieves a practical balance between efficiency and security for smart home IoT ecosystems, with a computational overhead of $<3 ms$ and a communication cost of 112 bytes per session, outperforming RSA-2048 and pairing-based schemes. Though its ECDLP-based server authentication and nonce-driven replay resistance provide robust defense against eavesdropping and replay attacks, the protocol’s reliance on insecure hashing for device authentication (Step 3) and lack of forward secrecy limit its resilience against key-compromise and impersonation threats. Compared to symmetric-key (HMAC-SHA256) and certificateless (Lightweight PKI) alternatives, the protocol’s asymmetric security foundation ensures stronger long-term trust but necessitates future integration of ECDSA signatures or ephemeral session keys to address authentication flaws and align with the forward secrecy guarantees of TinyECC. This positions the protocol as a viable candidate for energy-constrained deployments, pending enhancements to its device authentication mechanism.

References

1. Azrour M, Mabrouki J, Guezzaz A, Kanwal A. Internet of Things security: Challenges and key issues. Security and Communication Networks. 2021;2021:Article ID 5533843, 1–11. <https://doi.org/10.1080/09720529.2021.1957189>
2. Usman M, Ahmed I, Aslam MI, Khan S, Shah UA. SIT: A lightweight encryption algorithm for secure Internet of Things. International Journal of Advanced Computer Science and Applications. 2017;8(1).
3. Ehuil BB, Chen C, Wang S, Guo H, Liu J. A secure mutual authentication protocol based on visual cryptography technique for IoT-cloud. Chinese Journal of Electronics. 2024;33(1):43–57. <https://doi.org/10.23919/cje.2022.00.339>
4. Baccouri S, Farhat H, Azzabi T, Attia R. Lightweight authentication scheme based on Elliptic Curve El Gamal. Journal of Information and Telecommunications. 2023;8(2):231–261.
5. Hammi B, Fayad A, Khatoun R, Zeadally S, Begriche Y. A lightweight ECC-based authentication scheme for Internet of Things (IoT). IEEE Systems Journal. 2020;14(1):1–13. <https://doi.org/10.1109/JSYST.2020.2970167>
6. Baccouri, S., Farhat, H., Azzabi, T., & Attia, R. (2024). Lightweight authentication scheme based on Elliptic Curve El Gamal. *Journal of Information and Telecommunications*, *8*(2), 231-261.

 <https://doi.org/10.1080/24751839.2023.2281143>

1. Zhang Y, Chen Z, Wang X. Lightweight authentication for IoT devices using ECC. IEEE Internet of Things Journal. 2020;7(3):2254–2265.
2. Ahmed M, Zhang H, Li T. Blockchain-based mutual authentication for IoT systems. IEEE Transactions on Industrial Informatics. 2020;16(6):4146–4154.
3. Kumar S, Gupta R, Sharma N. Hash-based mutual authentication for IoT-cloud integration. IEEE Access. 2021;8:54321–54332.
4. Lee J, Kim H. Hybrid cryptographic methods for secure IoT communication. IEEE Sensors Journal. 2021;21(8):9483–9491.
5. Chen F, Wu L, Gao X. Dynamic key management for IoT authentication in cloud environments. IEEE Transactions on Cloud Computing. 2021;9(4):1457–1469.
6. Wang H, Zhao B, Liu F. Identity-based authentication scheme for IoT devices in cloud environments. Journal of Cloud Computing. 2022;11(1):112–128.
7. Singh V, Joshi R, Prakash A. Lightweight authentication using physically unclonable functions (PUFs) for IoT security. IEEE Transactions on Information Forensics and Security. 2020;15(6):1458–1472.
8. Rahman M, Islam N, Hossain M. AI-driven anomaly detection for mutual authentication in IoT-cloud systems. Computers & Security. 2023;121:103114.
9. Patel R, Mehta D, Bhatt P. Multi-factor authentication framework combining biometrics and ECC for IoT-cloud security. IEEE Transactions on Industrial Informatics. 2021;17(9):5743–5755.
10. Hussain S, Alam M, Raza H. Certificateless public key cryptography (CL-PKC) authentication for IoT-cloud systems. Future Internet. 2022;14(3):102.
11. Luqman M, Faridi AR. Authentication of fog-assisted IoT networks using Advanced Encryption credibility approach with modified Diffie–Hellman encryption. Concurrency and Computation: Practice and Experience. 2023;35(22):e7742. <https://doi.org/10.1002/cpe.7742>
12. Das S, Namasudra S. Lightweight and efficient privacy-preserving mutual authentication scheme to secure Internet of Things-based smart healthcare. Transactions on Emerging Telecommunications Technologies. 2023;34(11):e4716.
13. Alshawich I, Al-Haj A. An efficient mutual authentication scheme for IoT systems. The Journal of Supercomputing. 2022;78:16056–16087. <https://doi.org/10.1007/s11227-022-04520-5>
14. Hellman ME. An overview of public key cryptography. IEEE Communications Magazine. 2002;40(5):42–49.
15. Saqib M, Moon AH. A systematic security assessment and review of Internet of Things in the context of authentication. Computers & Security. 2023;125:103053.
16. Wang Q, Dong C, Jian S, Du D, Lu Z, Qi Y, Han D, Ma X, Wang F, Liu Y. HANDOM: Heterogeneous Attention Network Model for Malicious Domain Detection. Computers & Security. 2023;125:103059.
17. Yadav AK, Misra M, Liyanage M, Varshney G. Secure and user efficient EAP-based authentication protocol for IEEE 802.11 wireless LANs. In: 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS); 2020. p. 576–584.
18. Li M, Hu S. A Lightweight ECC-Based Authentication and Key Agreement Protocol for IoT with Dynamic Authentication Credentials. Sensors. 2024;24(24):7967. <https://doi.org/10.3390/s24247967>
19. Kokila M, Srinivasa Reddy K. Authentication, access control and scalability models in Internet of Things Security–A review. Cyber Security and Applications. 2025;3:100057.
20. Kirvan P. What is mutual authentication? Techtarget Network. 2023. Accessed 17 February 2025. Available: <https://www.techtarget.com/searchsecurity/definition/mutual-authentication>.
21. Ezugwu A, Ukwandu E, Ugwu C, Ezema M, Olebara C, Ndunagu J, Ofusori L, Ome U. Password-based authentication and the experiences of end users. Scientific African. 2023;21:e2019.
22. Monton AL. Certificate-based Authentication Explained. GlobalSign by GMO. 2021. Accessed 29 March 2025. Available: <https://www.globalsign.com/en/blog/sg/introduction-certificate-based-authentication>.
23. Leduc G. Verification of two versions of the challenge handshake authentication protocol (CHAP). Annals of Telecommunications. 2000;55:20–30. <https://doi.org/10.1007/BF02998328>
24. Chen JC, Wang YP. Extensible authentication protocol (EAP) and IEEE 802.1x: Tutorial and empirical experience. IEEE Communications Magazine. 2005;43(12):S26–S32.
25. Rachmayanti A, Wirawan W. Elliptic Curve Cryptography (ECC) Based Authentication Scheme on IoT Networks for Health Information Systems. In: 2024 International Seminar on Intelligent Technology and Its Applications (ISITIA); 2024. p. 125–130.
26. Jin X, Omote K. An efficient blockchain-based authentication scheme with transferability. PLOS ONE. 2024;19(9):e0310094. <https://doi.org/10.1371/journal.pone.0310094>
27. Thiyagarajan G. Enhancing Captive Portal Authentication with Zero-Knowledge Proofs (ZKP). International Journal of Computer Applications. 2023;186(48):43–51.
28. Alotaibi A, Aldawghan H, Aljughaiman A. A Review of the Authentication Techniques for Internet of Things Devices in Smart Cities: Opportunities, Challenges, and Future Directions. Sensors. 2025;25(16490):1–43.
29. Hu X, Li X, Zheng X, Liu Y, Xiong X. A high speed processor for elliptic curve cryptography over NIST prime field. IET Circuits, Devices & Systems. 2022;16(4):350–359. <https://doi.org/10.1049/cds2.12145>
30. Rabah K. Theory and Implementation of Elliptic Curve Cryptography. Journal of Applied Sciences. 2005;5(4):604–633.
31. Liu A, Ning P. TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In: Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN); 2008. p. 245–256.
32. Sekhar VC, Sarvabhatla M. Security in Wireless sensor networks with public key techniques. In: 2012 International Conference on Computer Communication and Informatics; 2012. p. 1–16.
33. Amrita CP, Ekwueme IH, Adam IH, Dwivedi A. Lightweight Cryptography for Internet of Things: A Review. EAI Endorsed Transactions on Internet of Things. 2024;10.
34. Shah A, Engineer M. A Survey of Lightweight Cryptographic Algorithms for IoT-Based Applications. In: Tiwari S, Trivedi M, Mishra K, Misra A, Kumar K, editors. Smart Innovations in Communication and Computational Sciences. Springer; 2019. p. 1–12.
35. Chen C, Guo H, Wu Y, Shen B, Ding M, Liu J. A Lightweight Authentication and Key Agreement Protocol for IoT-Enabled Smart Grid System. Sensors. 2023;23(8):3991. <https://doi.org/10.3390/s23083991>