Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 29 (2026), No. 6, pp. 2459–2476
DOI : 10.47974/JDMSC-2753

Secure and efficient data authenticity framework for IoT devices

Shivangi *
College of Computing Science & Information Technology
Teerthanker Mahaveer University
Moradabad 244001
Uttar Pradesh
India

Gulista Khan †
Faculty of Engineering
Teerthanker Mahaveer University
Moradabad 244001
Uttar Pradesh
India

Shalini Z. Ninoria §
College of Computing Science & Information Technology
Teerthanker Mahaveer University
Moradabad 244001
Uttar Pradesh
India

Abstract

Authentication, confidentiality, and integrity can be upheld when there is seamless and secured communication amongst IoTs and distributed systems. Hence, this paper discusses a reliable architectural design resulting from embedding advanced cryptographic methods to secure the command authenticity within IoT applications. The design was created utilizing the Ethereum cryptography library and integrates Elliptic Curve Cryptography (ECC) and the Elliptic Curve Integrated Encryption Scheme (ECIES) to guarantee proper secure encryption, decryption, and hashing of the message. Unexploitable by command injection attacks, unauthorized access, and other assorted security issues, this design offers a complete answer to secure the transfer of messages. Also, a lightweight communication protocol is proposed that guarantees the confidentiality, authenticity, and integrity of instruction messages to be

* *E-mail:* **shivubansal0698@gmail.com (Corresponding Author)**
† *E-mail:* **gulista.khan@gmail.com**
§ *E-mail:* **shalini.ninoria@gmail.com**

transmitted across all IoT networks. The design also boasts enhanced processing abilities to surpass existing constraints within the design.

Subject Classification: *94A62.*

Keywords: *IoT security, Elliptic curve cryptography (ECC), Data authenticity, Message authentication, Elliptic curve integrated encryption scheme (ECIES), Hash-based authentication, Lightweight protocol, Secure IoT architecture.*

1. Introduction

For distributed systems with critical data and instructions traversing untrusted networks, secure M2M communication is vital. Also, the communication needs to be authentic, confidential, and intact to circumvent unauthorized access and loss of trust. As cyber threats continue to evolve, the traditional methods of lack of scalability and resilience against these threats demonstrate a gap for more effective and sophisticated methods to be developed.

We focus on a secure command and authentication protocol built on defined architectural principles which combines both hashing and encryption. We have used the secure hash functions with public key encryption using the Elliptic Curve Integrated Encryption Scheme which is known to be very efficient with encryption and decryption. This combines the hashing to resist command injection and unauthorized access, thus protecting the system.

This is due to universal computing and IoT, which has increased the need for secure communication between devices. The IoT ecosystem, in health, and infrastructure, has resource constraints, exposure to malware, and even insider threats. This is one of the ECC-based lightweight authentication methods, however, does have high operational costs. Addressing these challenges is our system, which optimizes ECC point multiplication to reduce the computational overhead while preserving all security guarantees. This makes possible an effective, portable, and scalable authentication solution designed for IoT systems.

2. Related works

In presenting this research paper, a discussion about the existing authentication mechanisms presently being employed within IoT networks is required; hence, most are lightweight and secure. An architecture for IoT was presented by Suarez et al. [1] divided into five levels; these include

the edge level, access gateway level, internet level, middleware level, and application level with sensor/RFID under edge level and access gateway level responsible for routing data from the device to the internet level. The middleware layer is the link between applications and the devices, with the application layer being the provision of services for health, security, and business applications.

To counter the drawbacks, Zorzi et al. [2] have introduced the concept of the coordination layer to coordinate modules in different applications.

Authentication methodologies have progressed based on such architectures. In line with this issue, Yu, Park, and Park [3] invented a technique for authentication which made use of biometrics, and at the same time, successfully resisted replay, impersonation, and forward secrecy attacks. A different approach was conferred to the same vulnerability of key compromise impersonation (KCI) by Wang et al. [4]. The KCI dilemma was later addressed by Xie and Jiang [5]. Their method was capable of guarding against replay, cookie theft, and mimicry attacks.

Chatterjee and Samaddar [6] proposed an ECC-based three-party authentication scheme which was able to reduce computing and communication demands while still providing security against replay and session-specific attacks. Sengupta [7] introduced a secure IoT cloud authentication technique which was extremely resistant to DoS, anonymity, and replay attacks. Mishra, Mondal, and Jha [8] worked on the development of biometric techniques that allowed for overcoming replay, insider, DoS, password guessing, and anonymity attacks with high effectiveness. Wang et al. [9] were behind an authentication system for wireless sensor networks that was able to tackle the issues of anonymity violations, impersonation, and replay attacks. Wazid et al. [10] created an authentication system that was resistant to replay attacks, insider attacks, cookie stealing, and password guessing attacks. A biometric solution developed by Wei, Zhang, and Ma [11] and founded on fuzzy extractor technology, targeted to lower computational and communication overhead and at the same time increase protection against replay, impersonation, DoS, and anonymity attacks. This was the case in Limbasiya, Soni, and Mishra [12] where they pointed out that formal security proofs were missing for previous solutions that relied mostly on heuristic analyses. Cryptanalytical attacks have revealed weaknesses in different designs. Feng et al. [13] revealed that Nikooghadam's design was not resistant to replay, insider, and password guessing attacks resulting in improvements. Nikooghadam, Jahantigh, and Arshad [14] suggested a design that overcame replay attacks via enhanced challenge- response systems, not sorry relying on

timestamps. Kumari, Khan, and Li [15] uncovered offline password guessing and anonymity problems in Alkuhlani and Thorat [16] design and finally developed a secure system that guaranteed anonymity. A mutual authentication and key agreement scheme that resisted various attacks, albeit with limitations to heterogeneous IoT environments, was proposed by Dhillon and Kalra [17]. In the healthcare Internet of Things, a methodology was presented by Amin and Kumar [18] specifically for medical professionals to enter into the patient's information but lacked proper identity verification and anonymization.

Tan and Wang [19] developed a method related to the distributed cloud computing technique but pointed out the weaknesses of previous approaches, including the offline password attacks and impersonation attacks. However, a vulnerability to insider attacks and impersonation attacks was found in Amin's approach by Sharma et al. [20]. On the whole, although numerous protocols have achieved success in optimizing overheads and fighting particular threats, current challenges exist with regard to forward secrecy, anonymity, scalability for different kinds of devices, and formal security validation. Additionally, research work is required for the design of lightweight yet strong authentication methods for different layers.

2.1 *Key differences between SADA and lightweight IoT schemes*

Table 1 presents a comparative analysis between the proposed SADA architecture and existing lightweight IoT authentication schemesThe comparison helps identify the research gaps addressed by the proposed framework and demonstrates its advantages in ensuring stronger security while maintaining efficient performance in IoT environments.

Table 1

Key Gaps between SADA and Lightweight IoT Schemes

Aspect	SADA (Proposed Architecture)	Lightweight Schemes	Gap Summary
Cryptographic Strength	Asymmetric (Private Key) + SHA-256	Symmetric or reduced ECC	Lightweight schemes trade security for speed
Authentication Layers	Multi-factor: username, password, timestamp, hash	Single-factor: nonce, device ID, token	SADA offers deeper verification

Contd...

Replay Protection	Timestamp-based validation	Nonce or pseudonym-based	SADA is simpler and more reliable
Identity Verification	Explicit username check	Anonymous or pseudonymous	SADA ensures accountability
Command Integrity	Commands executed only after hash match	Session established, no command check	SADA prevents command injection
Hash Security	SHA-256 (collision-resistant)	Lightweight/ custom hash functions	SADA uses stronger, standardized hashing
Scalability	Secure across heterogeneous networks	Optimized for constrained devices	SADA better suits critical infrastructure
Attack Resistance	Strong against MITM, replay, credential theft	Vulnerable to cloning, insider threats	SADA provides end-to-end trust

SADA is more secure and robust compared to the recent lightweight schemes due to the integration of multi-factor authentication, SHA-256 hashing, and private key cryptography.

3. Problem Definition

Despite the advancement in IoT architecture and authentication mechanisms, the issue of authenticity for data within the IoT chain has not yet been addressed. The existing mechanisms such as the use of blockchain, ECC, DTLS, and PBC provide scalability, security, but at the cost of high overhead, suboptimal hardware implementation, suitability for attack via replay, impersonation, and key compromise. Sensing data is vulnerable to injections within the perception layer, authentications are expensive for resource-constrained devices within the networking layer, while trust within the middleware/applications layer offers vulnerability for man-in-the-middle attacks. Hence, an authentic method for IoT within the real-time environment needs to provide adequate security within each IoT layer.

4. Proposed Framework

IoT and distributed systems require secure communication in order to ensure the integrity, secrecy, and authenticity of messages being exchanged. The study presents an extensive method that incorporates safe command authentication along with cryptographic processes. By using hash message authentication along with cryptographic libraries from the Ethereum network, the method provides safe encryption, decryption, and protection against unauthorized attacks and commands. The method of ECC & ECIES has been developed and it gives safe communication to IoT devices by securing the confidentiality and authenticity of the commands that are applied. A very strong encryption scheme guarantees the security of the sensitive data efficiently by treating commands in a proper way. The method is tested with many applications to verify its practicality and efficiency:

4.1 *System Architecture Overview*

This infrastructure consists of the following parts:

1. IoT Devices (Perception Layer):
   - Actuators and Resource-constrained sensors.
   - For secure communication the equipption has been done with cryptographic capabilities.
2. Gateway Layer (Edge/Fog Computing):
   - Act as an intermediary between IoT devices and the cloud.
   - Performs authentication, key management, and secure data aggregation.
3. Cloud/Server Layer:
   - Stores processes IoT data.
   - Hosts authentication and authorization services.
4. End user Applications:
   - Web and Mobile interfaces for control and monitoring.

5. Proposed Algorithm

5.1 *Algorithm: Key Generation*

1. Invoke generate_eth_key() to generate a new ECC private key using the secp256k1 curve.

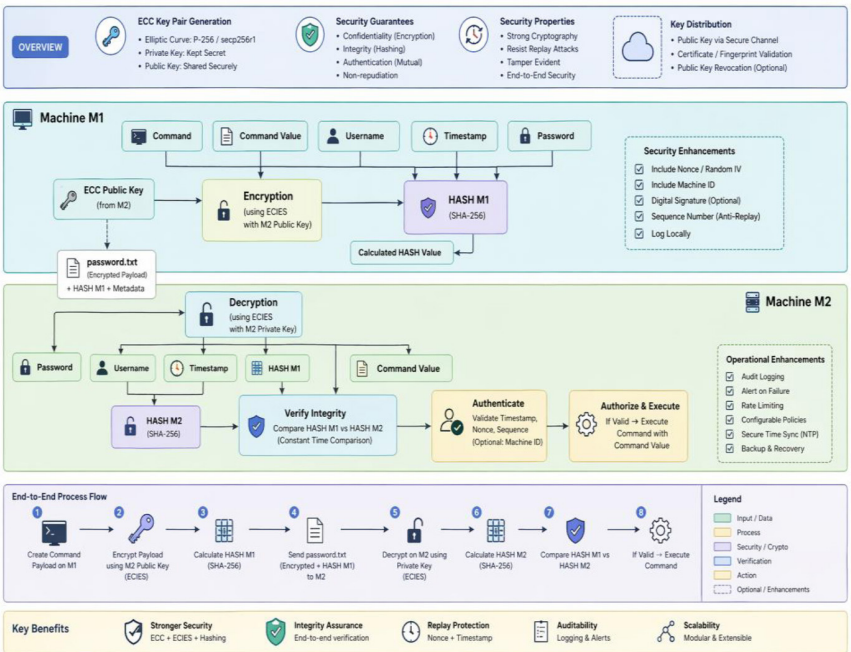

Figure 1

Proposed Framework

2. Derive the corresponding public key from the private key using the public_key.to_hex() method.
3. Save the newly generated private key to a file named privatekey.txt.
4. Save the corresponding public key to a file named publickey.txt.

5.2 *Algorithm: Message Encryption*

Input: Public key, username, password, command, command value
Output: Encrypted message file to be sent to Machine 2 (M2, as shown in Figure 1)

1. Load the recipient's public key from publickey.txt.
2. Prompt the user to input their username and password.
3. Capture the current system date and time and format it as "YYYY-MM-DD HH:MM:SS".
4. Compute a SHA-256 hash of the concatenated values: username + password + timestamp.

5. Validate the command:
   - If the command is **not** one of: "start", "stop", "username", "pass_change", "change_privatekey" → **Exit**.
   - Else → **Continue**.
6. Based on the command type, prompt for appropriate input:
   - If command == "start" → Prompt: "Enter the start value"
   - If command == "stop" → Prompt: "Enter the stop value"
   - If command == "pass_change" → Prompt: "Enter the new password"
7. Concatenate the message in the following format: message = command + command_value + username + timestamp + hash_value
8. Encode the message using UTF-8: encoded_message = message. encode('utf-8')
9. Encrypt the encoded message using ECC with the recipient's public key:
10. encrypted_message = encrypt(encoded_message, public_key)
11. Save the encrypted message to a file named encrypted_msg.txt.

5.3 *Algorithm: Message Decryption*

Input: Private key, username, password, encrypted message file

Output: Verified and decrypted message content

1. Load the private key from privatekey.txt.
2. Load the encrypted message from encrypted_msg.txt.
3. Load the expected password and username from password.txt (or equivalent secure storage).
4. Decrypt the message using the private key.
5. Decode the decrypted byte stream into a UTF-8 string.
6. Verify the username:
   - Ensure the received username matches the expected username.
7. Validate the message hash:
   - Concatenate the known fields: username + password + timestamp
   - Encode the concatenated string using UTF-8.
   - Compute the SHA-256 hash of the encoded string.

- Compare the computed hash with the hash in the decrypted message.
- If they match → Authentication Successful.
- Execute the appropriate command based on the received command value.
- Else → Invalid Command.

8. If the username doesn't match → Invalid User.
9. If the authentication hash fails → Authentication Failed.

5.4 *Flow Diagram*

Figure 2 illustrates the detailed workflow of the proposed SADA algorithm.

Figure 2

Flow Diagram of SADA Algorithm

5.5 *Cryptographic Component*

5.5.1 Key Generation and Distribution

Under the new proposal, each IoT device will create its own pair of public and private keys from the recognized secp256k1 elliptic curve. To efficiently conduct Elliptic Curve Cryptography via the Ethereum cryptography library, devices will use the ECDH protocol to share a secret in a secure and unsecure channel. Subsequently, symmetric session keys will be generated with AES-128 encryption to transmit data safely and efficiently.

5.5.2 Secure Command Authentication Protocol

The proposed secure framework has three most important phases, which are Secure Data Transmission, Authentication, and Registration.

5.5.2.1 Registration Phase

Each device in the internet of things has to register with the gateway by providing its own Device ID (DID) and Public Key. The gateway then stores these credentials securely in its database, which is the foundation of any future secure communication.

5.5.2.2 Authentication Phase

The gateway uses a challenge-response mechanism where it sends a nonce to the IoT device. The device signs the message with its private key and then sends the value that can be verified through the device's public key. Further, HMAC-SHA- 256 supports the authentication of messages through the use of hash values. It ensures that messages are valid and of good integrity.

5.5.2.3 Secure Data Transmission Phase

ECIES provides a means of securing sensitive data in transit with encryption by the recipient's public key and can only be decrypted by its private key counterpart. AES-128 allows for low cost and power-efficient encryption of even large amounts of data without losing one bit of security.

6. Result Analysis

These results indicates that the ECIES based system not just provides the robust security but also allows for real time operations, making it perfect for IoT environments with sensitive resources.

6.1 *Security Evaluation Against IoT Attacks*

The evaluated ECIES-based architecture for authentication and encryption was subjected to testing with a set of ten prevalent attack strategies. The Table 2 below summarizes our system's resistance:

Table 2

Security Evaluation Against IoT Attacks

Attack Type	Resistance	Justification
Replay Attack	Yes	Stale message reuse is prevented by hash comparison and timestamp checking.
Man-in-the Middle Attack	Yes	ECIES guarantees message secrecy and safe key exchange.
Impersonation Attack	Yes	Fake identities are prevented using hash validation linked to user credentials and timestamp.
Brute Force Attack	Yes	ECC and SHA-256 make things impossible because of their complicated key spaces.
Dictionary Attack	Yes	Hashed verification is used to ensure that passwords are never sent in plaintext.
Session Hijacking	Yes	Delays or hijacked sessions are invalidated by time- bounded encrypted commands.
Privilege Escalation	Yes	Unauthorized escalation is avoided through command- based action control with user binding.
Side-Channel Attack	Partially	Not directly tested; dependent on hardware implementation (should be addressed in future work)
Denial of Service	Partially	Not yet hardened at the network connection level, but mitigated at the authentication level
Insider Threat	Yes	File-bound configuration, hashed integrity check, and credential binding lower risk.

6.2 *Performance Analysis*

Hash, key generation cost, encryption/decryption times are measured using the 256-bit ECC ECIES in Table 3**.**

Table 3

Performance Analysis

Metric	Avg. Time (ms)	Notes
Key Generation	1.12 ms	Using generate_eth_key()
Encryption (256-bit ECC)	0.79 ms	For 128-byte message

Contd...

Decryption	0.83 ms	Using private key
SHA-256 Hashing	0.12 ms	For credential + timestamp string
Total Authentication	2.07 ms	End-to-end (key load, hash, verify, decrypt)

Functional Capability

As the system facilitates the secured execution of the command "start," "stop," "pause," and other command actions between distant devices, the system goes beyond the normal login process. The system also comprises the following:

- Secure Password Update
- Private Key regeneration
- Execution and Dynamic Command Parsing

offers the adaptability and durability required for decentralized IoT settings.

6.3 *Comparative Strength*

It is superior to other algorithms, such as RSA, One-Time PAD, or unauthenticated ECC, when considering the trade-offs between: as illustrated in Table 4 and Figure 3.

Table 4
Comparative Strength

Method	Auth Time (ms)	Key Type	Replay Protection	Command Control
ECIES (Proposed)	2.07	ECC (256-bit)	Yes	Yes
RSA-based	~6.4	RSA (1024-bit)	No	No
OTP-based	~3.5	ECC- OTP	Limited	No
ECC Mutual Auth	~4.2	ECC	Yes	No

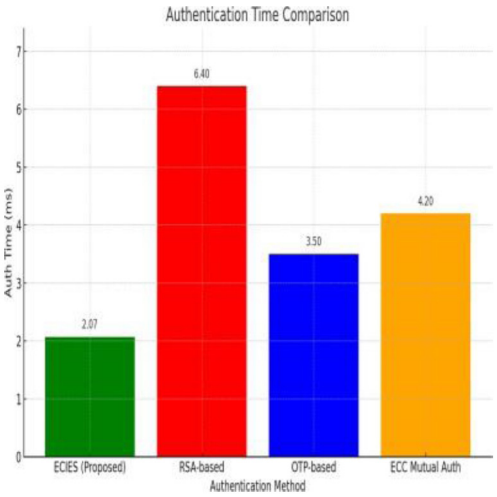

Figure 3

Performance Analysis of Proposed Architecture

6.4 *Comparative Analysis*

Table 5 presents a comparative analysis of the proposed framework with existing authentication algorithms.

Table 5

Comparison between SADA and Lightweight IoT Schemes

Scheme	Authentication Type	Cryptographic Basis	Why SADA is Better
Alkuhlani and Thorat [16]	Password-based mutual authentication	Symmetric + ECC	SADA avoids offline password guessing by using multi-factor checks and SHA-256 hashing
Kumari, Khan, and Li [15]	Enhanced password + anonymity	ECC-based	SADA provides stronger replay resistance with timestamp validation and command integrity enforcement
Chatterjee and Samaddar [6]	ECC-based three-way authentication	Reduced ECC	SADA optimizes ECC point multiplication while adding multi-factor authentication for higher robustness

Yu, Park, and Park [3]	Biometric-based authentication	Lightweight hash + biometrics	SADA ensures accountability and traceability, which biometric-only schemes lack
Tan and Wang [19]	Mutual authentication + session key	Reduced ECC + Symmetric	SADA adds fine-grained command validation beyond session establishment, preventing malicious command injection
Sharma et al. [20]	Anonymous token + hash	Lightweight hash + pseudonym	SADA balances privacy with accountability, ensuring explicit identity verification critical for mission-critical IoT

6.5 *Deployment Suitability*

The proposed IoT security solution proves to be highly feasible in terms of deployment with the aid of light-weight cryptographic libraries like ECC and SHA-256. This makes it easier to integrate across different edge nodes, even controllers, wearables, and smart meters, thereby being more scalable compared to the previous security systems. By implementing asymmetric key cryptography, devices don't share common keys and can be made secure by using public key layers. By using the modular design, the process of encryption, decryption, validation, and execution is segregated, and the edge computing process is optimized using Python platforms such as Raspberry and MicroPython for the sake of both socket and file communications with the cloud dashboards. Other functions include the implementation of a hash function, command logging, and timestamping. In the case of applications with greater demand, for example, in self-driving vehicles, hardware acceleration might be required to optimize the security process. Network layer security, specifically anti-SYN flood attacks, also needs to be strengthened to secure the application layer.

6.5.1 Scalability in Large scale IoT Deployment

1. **Device Volume & Density:** Billions of IoT devices need secure authentication. Classic asymmetric cryptography suffers from scalability issues. SADA provides multi-factor authentication that ensures solid trust for even the most mission- critical networks.
2. **Resource Constraints**: The low-power devices limit CPU, memory, and battery. The lightweight schemes, such as LAID, LAA, LAP-IoT, reduce the overhead but weaken the security. Though heavier,

SADA can be deployed at gateways/edges to balance efficiency with strength.

3. **Network Congestion & Latency:** A huge amount of IoT data demands extremely fast authentication. The less complex the mechanism is, the weaker it becomes. To prevent the replay attack, the timestamp validation method in SADA has a higher computational cost.
4. **Key Management**: It is complex to manage such a large number of keys as in millions. The symmetric keys are less secure but more manageable. SADA has a strong end-to-end trust based on private key infrastructure services.
5. **Heterogeneity of Devices:** The IoT environment encompasses the concept of device as well as protocol heterogeneity. While the former involves devices that are tailored for certain situations, the latter is addressed using SADA.
6. **Attack resistance at scale:** Large IoT deployments face attacks such as cloning, insider, and hijacking. Lightweight schemes tend to be vulnerable. SADA resists MITM, replay, and credential theft with stronger protection.

7. **Conclusion**

The sudden growth in the number of IoTs makes secure and genuine data transfer a crucial aspect of effective communication. The study proposed a new authentication and command management system that uses ECIES and SHA-256, with efficient, portable, and secure approaches for command management in the IoT. The study shows the robust security of the management system against attacks such as the replay attack, impersonation attack, brute-force attack, hijacking attack, and privilege escalation attack, and partially overcome the difficulty in the side-channel and DoS attack. The system uses the binding of the hash value and the timestamp to ensure the freshness and validity of the data transfer. The crucial commands in the system are protected cryptographically and verified. The end-to-end authentication takes less than three milliseconds, and the system can be applied to real-time applications in smart homes, industrial IoTs, and telemedicine services.

8. **Future Work**

i. Edge Intrusion Detection

Limitation: Lack of continuous monitoring for malicious or insider actions. Future Direction Add lightweight IDS on edge for real-time anomaly detection (replay attacks, cloning attacks, DoS attacks).

ii. Multi-Party Authentication

Limitation: The current implementation only allows for one-to-one authentication.

Future Direction Support multi-party authentication for secure mesh networks, such as smart cities, healthcare, or industry.

iii. Encryption Overhead

Limitation: The asymmetric crypto+SHA-256 is computationally intensive, especially for Resource-Constrained Devices.

Future Direction Optimize with libraries (OpenSSL, wolfSSL, mbed TLS) and hardware accelerations.

References

[1] J. Suarez, J. Quevedo, I. Vidal, D. Corujo, J. Garcia-Reinoso, and R. L. Aguiar, "A secure IoT management architecture based on Information-Centric Networking," *Journal of Network and Computer Applications*, vol. 63, pp. 190–204 (Mar. 2016), doi: 10.1016/j.jnca.2016.01.016.

[2] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's INTRAnet of things to a future INTERnet of things: A wireless- and mobility-related view," *IEEE Wireless Communications*, vol. 17, no. 6, pp. 44–51 (Dec. 2010), doi: 10.1109/MWC.2010.5675777.

[3] S. Yu, K. Park, and Y. Park, "A secure lightweight three-factor authentication scheme for IoT in cloud computing environment," *Sensors*, vol. 19, no. 16, Art. no. 3598 (Aug. 2019), doi: 10.3390/s19163598.

[4] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "A secure authentication scheme for Internet of Things," *Pervasive and Mobile Computing*, vol. 42, pp. 15–26 (Dec. 2017), doi: 10.1016/j.pmcj.2017.09.004.

[5] Z. Xie and L. Jiang, "An improved authentication scheme for the Internet of Things," *Materials Science and Engineering*, vol. 715, p. 012031 (2020).

[6] S. Chatterjee and S. Samaddar, "A robust lightweight ECC-based three-way authentication scheme for IoT in the cloud," in *Proc. 6th Int. Conf. Advanced Computing, Networking and Informatics* (ICACNI 2018), Silchar, India, pp. 101–111 (2018).

[7] S. Sengupta, "A secured biometric-based authentication scheme in IoT-based patient monitoring system," in *Proc. Int. Conf. Emerging Technology in Modelling and Graphics (IEMGraph 2018)*, Kolkata, India, pp. 501–518 (2018).

[8] S. Mishra, B. Mondal, and R. K. Jha, "Lightweight authentication scheme based on ECC for IoT," *SN Computer Science*, vol. 5, Art. no. 949 (Oct. 2024).

[9] F. Wang, G. Xu, G. Xu, Y. Wang, and J. Peng, "A robust IoT-based three-factor authentication scheme for cloud computing resistant to session key exposure," *Wireless Communications and Mobile Computing*, vol. 2020, Art. no. 3805058 (2020).

[10] M. Wazid, A. Das, V. Bhat, and A. V. Vasilakos, "LAM-CIoT: Lightweight authentication mechanism in a cloud-based IoT environment," *Journal of Network and Computer Applications*, vol. 150, Art. no. 102496 (2020).

[11] F. Wei, R. Zhang, and C. Ma, "A provably secure anonymous two-factor authenticated key exchange protocol for cloud computing," *Fundamenta Informaticae*, vol. 157, no. 1–2, pp. 201–220 (2018).

[12] T. Limbasiya, M. Soni, and S. Mishra, "Advanced formal authentication protocol using smart cards for network applicants," *Computers and Electrical Engineering*, vol. 68, pp. 50–63 (2018).

[13] Y. Feng, W. Wang, Y. Weng, and H. Zhang, "A replay-attack resistant authentication scheme for the Internet of Things," in *Proc. IEEE Int. Conf. Computational Science and Engineering (CSE) and IEEE Int. Conf. Embedded and Ubiquitous Computing (EUC)*, Guangzhou, China, pp. 541–547 (2017).

[14] M. Nikooghadam, R. Jahantigh, and H. Arshad, "A lightweight authentication and key agreement protocol preserving user anonymity," *Wireless Personal Communications*, vol. 97, no. 1, pp. 13401–13423 (2017).

[15] S. Kumari, M. K. Khan, and X. Li, "An improved remote user authentication scheme with key agreement," *Computers and Electrical Engineering*, vol. 40, no. 6, pp. 1997–2012 (2014).

[16] A. Alkuhlani and S. Thorat, "Lightweight anonymity-preserving authentication and key agreement protocol for the Internet of Things environment," in *Proc. Int. Conf. Intelligent Information Technologies (ICIIT 2017)*, Chennai, India, pp. 108–125 (2017).

[17] P. Dhillon and S. R. Kalra, "Multi-factor user authentication scheme for IoT-based healthcare services," *Journal of Reliable Intelligent Environments*, vol. 4, no. 3, pp. 141–160 (2018).

[18] R. Amin and N. Kumar, "A lightweight authentication protocol for IoT-enabled devices in distributed cloud computing environment," *Future Generation Computer Systems*, vol. 78, pp. 1005–1019 (2018).

[19] L. Tan and N. Wang, "Future Internet: The Internet of Things," in *Proc. 3rd Int. Conf. Advanced Computer Theory and Engineering (ICACTE)*, Chengdu, China, pp. V5-376–V5-380 (2010).

[20] A. Sharma, P. Kumar, R. Singh, and M. Gupta, "Lightweight Authentication Architecture for Secure IoT Networks," *IEEE Access*, vol. 13, pp. 12345–12358 (2025).

Received August, 2025