A Secure Authentication Protocol for IoT Devices

Mohamed Ech-Chebaby, Hicham Zougagh, Hamid Garmani, Zouhair Elhadari
Faculty of Sciences and Techniques, Moulay Slimane University, Beni Mellal, Morocco

***Abstract*—The rapid evolution of the Internet of Things (IoT) offers vast opportunities in automation and connectivity, yet simultaneously introduces critical security challenges. One of the most pressing concerns lies in the heterogeneity and limited computational capabilities of IoT devices, which complicate the deployment of robust security mechanisms. In this work, we present a lightweight and secure authentication protocol designed to establish mutual authentication between a server and smart objects. Our protocol enhances the scheme proposed by Fatma et al., addressing its identified vulnerabilities. Formal security analysis using AVISPA and ProVerif confirms the protocol's resilience against a wide range of threats. Furthermore, a practical simulation was conducted using a Raspberry Pi as the IoT device and a Core i5-based server to evaluate real-world performance. Results show that the protocol executes efficiently in real-time with a reduced authentication delay, demonstrating its feasibility for resource-constrained environments. This research contributes to the development of effective, scalable, and secure authentication solutions tailored for the IoT landscape.**

***Keywords*—IoT; Internet of Things; security; authentication**

I. Introduction

The widespread adoption of the Internet of Things (IoT) has transformed the way devices interact and exchange information, leading to a substantial increase in the number of connected smart devices worldwide. This unprecedented growth has amplified the demand for robust security mechanisms, particularly for mutual authentication between devices and servers. In such distributed environments, it is essential to guarantee not only reliable communication but also the confidentiality, integrity, and authenticity of the exchanged data [1], [2], [3].

Authentication is a cornerstone of IoT security, as it establishes trust between communicating entities before any sensitive information is exchanged. Over the years, numerous authentication protocols have been developed to secure the establishment of sessions between servers and smart objects. However, extensive analysis of existing schemes has revealed that many of them remain vulnerable to critical threats such as impersonation, replay, and man-in-the-middle (MitM) attacks. These weaknesses can be exploited by malicious actors to compromise both the confidentiality and integrity of IoT communications, thereby undermining the overall security of the system [4], [5].

In particular, the protocol proposed by Fatma et al. [6], while offering lightweight computation, exhibits several vulnerabilities that may lead to the disclosure of sensitive credentials and session keys. Such flaws pose serious risks in IoT scenarios, especially in resource-constrained environments where devices cannot easily implement computationally intensive countermeasures.

To address these limitations, this study introduces an enhanced authentication protocol designed for IoT environments. Our scheme leverages elliptic curve cryptography (ECC) for efficient key exchange and collision-resistant hash functions for identity protection and message integrity. ECC offers a favorable trade-off between computational efficiency and cryptographic strength, making it especially suitable for devices with constrained processing power, memory, and energy resources. The proposed design incorporates random nonces, key-derived masking, and challenge-response mechanisms to ensure resilience against known attack vectors while maintaining low overhead.

The security of the proposed protocol has been rigorously evaluated using formal verification methods to identify and mitigate potential vulnerabilities before deployment. Simulation and analysis demonstrate that our protocol achieves the following key security properties:

- Mutual authentication between server and device.
- Resistance to impersonation, replay, and MitM attacks.
- Confidentiality of transmitted data and integrity of messages.
- Low computational and communication cost suitable for IoT devices.

By integrating these measures, the proposed protocol provides a significant improvement in both security and operational efficiency over existing schemes. Ultimately, this work aims to contribute to the development of practical, scalable, and secure authentication solutions for the IoT ecosystem, ensuring trustworthy communication even in highly resource-constrained environments.

II. The Mechanisms used by Authentication Protocols

A. Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is a well-established cryptographic method known for providing robust security in authentication schemes [7]. It is based on the mathematical structure of elliptic curves, typically represented as:

$$y^2 = x^3 + ax + b \tag{1}$$

where, a and b are curve parameters. In ECC, cryptographic operations involve algebraic manipulations on points belonging to the elliptic curve. For example, adding two points P and Q on the curve yields a third point R, while multiplying a point P by a scalar k produces another valid point on the curve [8].

A key advantage of ECC is that it achieves comparable security to other public-key systems with significantly smaller

key sizes. For instance, a 128-bit ECC key can offer a security level equivalent to that of a 1024-bit RSA key. This property reduces both computational overhead and storage requirements, making ECC highly suitable for resource-constrained devices [8].

The security of ECC is rooted in the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP), which involves determining the integer k such that $kP = Q$, where P and Q are known points on the curve. Solving this problem is computationally infeasible for appropriately chosen curve parameters and key sizes [9], [10].

Due to its combination of mathematical robustness, efficiency, and reduced key size requirements, ECC is extensively employed in modern authentication protocols to provide secure entity verification and protect the confidentiality of exchanged information [11].

B. Cryptographic Hash Function

A cryptographic hash function is a mathematical algorithm that converts input data of arbitrary length into a fixed-length output, commonly referred to as a hash or message digest [12]. Such functions are fundamental in numerous areas of computer science, particularly for ensuring data security and verifying information integrity, as illustrated in the following Fig. 1.

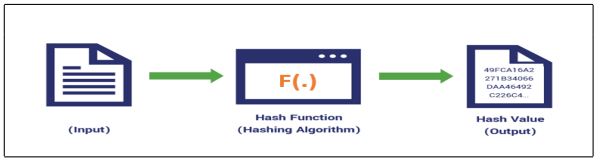

Fig. 1. Cryptographic hash function.

The hash functions possess the following properties:

- Deterministic: A cryptographic hash function is deterministic if the same input consistently yields the same hash value.
- Collision resistance: A hash function has collision resistance when it is computationally infeasible to find two distinct inputs that produce the identical hash output.
- Irreversibility: A hash function is irreversible if recovering the original input from its hash value is computationally impractical or impossible.

The most commonly used cryptographic hash functions include MD5, SHA-1, SHA-256, and SHA-3.

MD5, once widely adopted, is now considered insecure due to identified vulnerabilities and should not be used for security-critical purposes [13].

Similarly, SHA-1, although still present in some systems, is regarded as weak against modern cryptographic attacks [14].

In contrast, SHA-256 and SHA-3 are currently preferred for contemporary applications, as they provide stronger resistance to known attack methods [15].

III. Related Works

Authentication protocols in the Internet of Things (IoT) domain are a frequent subject of discussion in the scientific literature and have recently attracted considerable attention.

In [16], the authors propose a secure multifactor authentication scheme for cloud-based systems, demonstrated in the context of electronic health records. The approach combines three authentication factors—something the user knows, has, and is with cryptographic techniques such as RSA digital signatures and hashed credentials to achieve mutual authentication and ensure data integrity. Key features include an anonymous health center, acting as a trusted third party, responsible for distributing secret keys and managing user permissions during the registration phase. Additionally, the scheme incorporates a QR code mechanism to securely transmit doctor-signed electronic prescriptions to pharmacists. The protocol was implemented with entities such as patients, doctors, and a hospital server, and its security was validated using both the Scyther formal verification tool and informal analysis, demonstrating resistance to man-in-the-middle, replay, impersonation, insider, and phishing attacks.

Boonkrong's work [17] revisits Park et al.'s [18] multifactor biometric remote user authentication scheme and exposes several critical weaknesses, then offers an enhanced protocol to resolve them. Park et al.'s scheme – an improvement over earlier methods – was found to lack adequate message integrity protection (allowing an attacker to modify messages without detection), and freshness guarantees (making replay attacks feasible). It also did not implement a true challenge–response mechanism, leading to incomplete mutual authentication and the possibility of man-in-the-middle attacks, and it provided no way for the client to prove to the server that both share the same session key (leaving session key agreement only partially verified).

In [19], the authors addressed the challenge of safeguarding large volumes of sensitive user data in electronic healthcare (e-health) systems, particularly when using wireless devices with limited processing and storage capabilities. They proposed a dynamic privacy-preserving approach that enables server-side biometric authentication while ensuring complete user anonymity. In their method, the server does not have access to the exact biometric template value, which mitigates the risk of privacy leakage. Moreover, all messages transmitted through their scheme are untraceable, thereby maintaining a high degree of anonymity.

In [20], the researchers focused on the design of an end-to-end mutual authentication protocol for Wearable Health Monitoring Systems (WHMS). They analyzed the mutual authentication protocol proposed by Amin et al. [21] and identified several security flaws. Their improved version, based on quadratic residues, mitigates vulnerabilities such as stolen mobile device attacks, de-synchronization attacks, and sensor key exposure.

The work in [22] addressed the development of lightweight security mechanisms for Wireless Body Area Networks (WBANs), with an emphasis on securely transmitting sensitive patient information. The authors evaluated a lightweight authentication protocol proposed by Liu et al. [23] and, through a security assessment, found weaknesses in its design.

To overcome these limitations, they proposed a single-round lightweight authentication scheme that enhances security and reduces computational overhead.

In [6], an authentication protocol was designed for IoT-based healthcare applications. The authors introduced two major enhancements to the recently proposed M2C mutual authentication protocol, which was originally intended for RFID-based healthcare systems. Their modified protocol, referred to as M2M, leverages Elliptic Curve Cryptography (ECC) to secure RFID communications in healthcare environments.

The authors of [24] proposed an authentication mechanism for Medical Body Area Networks (MBANs) based on patient body motion. They developed a generalized model capable of characterizing routine patient activities—such as walking and running—to verify the legitimacy of sensor nodes. Their security analysis demonstrated that the scheme is resistant to well-known attack vectors.

Using ECC, [25] proposed an RFID-based mutual authentication scheme aimed at improving patient medication safety. The scheme enables secure exchanges between an RFID tag and a medication server, ensuring reliable medical evidence for prescriptions and dosage administration. Nevertheless, this protocol was later shown to have vulnerabilities, which were addressed by Fatma et al. [6].

In [26], the authors presented an authentication protocol for IoT-based RFID systems, focusing on mitigating security risks inherent in wireless communication channels. Their design integrates ECC and employs the elliptic curve Diffie–Hellman (ECDH) [27] key agreement mechanism to generate temporary shared keys for encrypting transmitted messages.

Several studies have also presented mutual authentication schemes aimed at establishing a secure session key between RFID tags and backend servers to ensure reliable communication. Examples include the works of Dinarvand et al. [28], Liao et al. [29] for specialized environments, and Zhao et al. [30] for healthcare-specific applications.

All these schemes generally follow a two-phase structure. The configuration phase involves the server generating ECC public/private key pairs to be used in the subsequent stage. The authentication phase is then repeated each time the server initiates a connection with a smart object, during which both entities exchange messages to mutually verify their identities.

IV. Description and Notation Used

Before presenting our enhanced version of the M2C authentication protocol derived from the scheme proposed by Fatma et al. [6], Table I summarizes the notations employed in its description.

V. Review and Cryptanalysis of Fatma et al.'s Protocol

Fatma et al. [6] introduced an ECC-based authentication scheme tailored for resource-limited devices, including RFID tags and sensors. Owing to its design, it falls under the category of machine-to-cloud (M2C) protocols. The proposed approach functions through two primary phases.

TABLE I. Notations used in the Protocol Description (Modified Symbols)

Notation	Meaning
n_{tot}	Total of IoT devices.
$\mathcal{S}$	Server in Cloud.
$\mathcal{D}$	IoT device.
D_k	Specific IoT device D_k, where $k \in [1, n_{tot}]$
q_a, q_b	Two large prime numbers.
$\mathbb{F}^*_{q_a}$	Multiplicative group of a finite field.
$\mathcal{E}$	Elliptic curve defined by $y^2 = x^3 + ax + b$.
P_g	Base point (generator) of $\mathcal{E}$ of order q_a.
(q_b, a, b, q_a, P_g)	Domain parameters for constructing $\mathcal{E}$.
H	Collision-resistant secure hash function.
$\rho \xleftarrow{\$} \mathbb{F}^*_{q_a}$	Random element ρ sampled from $\mathbb{F}^*_{q_a}$.
$\alpha \cdot \beta$	Scalar multiplication of α with elliptic curve point β.
$\alpha \oplus \beta$	XOR operation.
$\alpha \parallel \beta$	Concatenation operation of α and β.
$\alpha \stackrel{?}{=} \beta$	Equality check.
ID_{D_k}	Unique identity of device D_k.
hID_{D_k}	Hash of the identity ID_{D_k} stored in the server database.
$\mathsf{SK}_{\mathcal{S}}$	Server's private key.
$\mathsf{PK}_{\mathcal{S}}$	Server's public key.

A. The Configuration Phase

In this phase, the server S generates the system parameters (q_b, a, b, q_a, P_g) required for the construction of the elliptic curve $\mathcal{C}$, shares them with other entities, and stores them in its database. The server then assigns to each smart object D_k, a unique identifier ID_{D_k}, and computes its hashed form $hID_{D_k} = \mathsf{H}_c(ID_{D_k})$, where hID_{D_k} serves as the key in the hash table used to store the identifiers of all smart objects. This process is illustrated in Fig. 2.

Server $\mathcal{S}$	**IoT Device** D_k
$(q_b, a, b, q_a, P_g) \xleftarrow{\$}$	
$ID_{D_k} \xleftarrow{\$}$	
$hID_{D_k} = \mathsf{H}(ID_{D_k})$	
$\xrightarrow{(q_b, a, b, q_a, P_g),\ ID_{D_k},\ hID_{D_k}}$	

Fig. 2. Configuration phase.

B. The Authentication Phase

This step is illustrated in Fig. 3 and proceeds as follows:

- The server $\mathcal{S}$ randomly selects $\rho_{\mathcal{S}} \in \mathbb{F}^*_{q_a}$ and computes $R_{\mathcal{S}} = \rho_{\mathcal{S}} \cdot P_g$. It then transmits $R_{\mathcal{S}}$ to the device D_k.
- Upon reception, D_k picks a random $\rho_k \in \mathbb{F}^*_{q_a}$ and calculates $R_k = \rho_k \cdot P_g$. To conceal hID_{D_k}, it derives a shared key $K = \rho_k \cdot R_{\mathcal{S}}$ and masks the identity hash as $M = hID_{D_k} \oplus K$. It then generates an authenticator $\mathsf{Auth}_k = \mathsf{H}(ID_{D_k} \parallel 2 \parallel K)$. The device sends the tuple $(R_k, \mathsf{Auth}_k, M)$ to $\mathcal{S}$.
- On receiving the message, $\mathcal{S}$ computes $K = \rho_{\mathcal{S}} \cdot R_k$ and retrieves $hID'_{D_k} = M \oplus K$. It searches the

database for the identity ID'_{D_k} corresponding to hID'_{D_k}. If no match is found, $\mathcal{S}$ ends the session after a predefined timeout to limit information leakage. Otherwise, it computes $\mathsf{Auth}'_k = \mathsf{H}(ID'_{D_k} \parallel 2 \parallel K)$ and compares it to the received Auth_k. If they differ, the session is aborted; if equal, D_k is authenticated. The server then generates $\mathsf{Auth}_{\mathcal{S}} = \mathsf{H}(ID_{D_k} \parallel 1 \parallel K)$ and sends it to D_k.

- Upon receiving $\mathsf{Auth}_{\mathcal{S}}$, the device computes $\mathsf{Auth}'_{\mathcal{S}} = \mathsf{H}(ID_{D_k} \parallel 1 \parallel K)$ and checks equality. If they match, the server is authenticated, achieving mutual authentication; otherwise, the session is terminated.

At the end of this phase, D_K and S share the session key, K, which has been computed to ensure integrity, privacy, and confidentiality during subsequent exchanges.

VI. Potential Security Weaknesses in the Fatma et al. Protocol

In the previous scheme, an adversary is able to recover the communication key K, as well as the identifier ID_{D_K} of a smart object, enabling them to intercept exchanges between the device D_K and the server $\mathcal{S}$. The attacker first obtains the hashed identifier H_{D_K} by following the steps illustrated in Fig. 4:

- The adversary $\mathcal{A}$ selects a random value $\rho_{\mathcal{A}}$ and computes $R_{\mathcal{A}} = \rho_{\mathcal{A}} \cdot P_g$. It then sends $R_{\mathcal{A}}$ to the device D_k.
- Upon receiving $R_{\mathcal{A}}$, D_k generates a random $\rho_k \in \mathbb{F}^*_{q_a}$ and computes $R_k = \rho_k \cdot P_g$. To protect hID_{D_k}, the device calculates $K = \rho_k \cdot R_{\mathcal{A}}$ and masks it as $M = hID_{D_k} \oplus K$. It also generates $\mathsf{Auth}_k = \mathsf{H}(ID_{D_k} \parallel 2 \parallel K)$, and sends $(R_k, \mathsf{Auth}_k, M)$ to $\mathcal{A}$.
- The adversary then computes $K = \rho_{\mathcal{A}} \cdot R_k$ and retrieves hID_{D_k} by computing $hID_{D_k} = K \oplus M$.

After the recovery of hID_{D_k}, the attacker can perform the following two attacks:

A. Eavesdropping Attack

Once the adversary has obtained hID_{D_k}, it stores this value in its own database. During a legitimate authentication process between $\mathcal{S}$ and D_k, the adversary can perform an eavesdropping attack to derive the session key K, as outlined in Fig. 5:

- The server $\mathcal{S}$ randomly selects a number $\rho_{\mathcal{S}}$ and computes $R_{\mathcal{S}} = \rho_{\mathcal{S}} \cdot P_g$. It then transmits $R_{\mathcal{S}}$ to D_k.
- The IoT device executes its computations and returns the tuple $(R_k, \mathsf{Auth}_k, M)$ to $\mathcal{S}$.
- The adversary intercepts this message. With knowledge of hID_{D_k} and access to $(R_k, \mathsf{Auth}_k, M)$, it can compute the communication key as $K = hID_{D_k} \oplus M$.

B. Preimage Attack

Once hID_{D_k} has been recovered, note that $hID_{D_k} = \mathsf{H}(ID_{D_k})$. Since ID_{D_k} is a scalar, a high-performance computing system could, in principle, generate candidate scalars x_k and compute $\mathsf{H}(x_k)$ until finding some x_n such that $\mathsf{H}(x_n) = \mathsf{H}(ID_{D_k})$. In that case, x_n reveals ID_{D_k}.

If ID_{D_k} is chosen with a sufficiently large bit length, such a brute-force search becomes computationally infeasible. However, this mitigation is constrained in IoT settings, where limited device storage makes it impractical to significantly increase the

VII. The Proposed ECC based Authentication Protocol

We propose an enhanced version of the protocol introduced by Fatma et al. [6] in order to address the vulnerabilities identified in Section VI. The improved scheme also supports mutual authentication between servers and multiple smart objects, provided that the servers store the identities of these objects.

The proposed protocol operates in two main phases: a configuration phase and an authentication phase.

A. The Configuration Phase

The server first creates the system parameters (q_b, a, b, q_a, P_g) and shares them with the other entities to construct the elliptic curve $\mathcal{E}$. Then, the server $\mathcal{S}$ chooses a random value $\mathsf{SK}_{\mathcal{S}} \in \mathbb{F}^*_{q_a}$ as its private key and computes its public key $\mathsf{PK}_{\mathcal{S}} = \mathsf{SK}_{\mathcal{S}} \cdot P_g$. Afterwards, the server generates and sends to each IoT device an identifier ID_{D_k} and $hID_{D_k} = \mathsf{H}(ID_{D_k})$. It also transmits its public key $\mathsf{PK}_{\mathcal{S}}$. Once this phase is completed, the server records the system parameters along with the device identifiers in its database, to be used later during the authentication phase (see Fig. 6).

B. Authentication Phase

This phase consists of three message exchanges, illustrated in Fig. 7.

- Challenge from server to device: The server $\mathcal{S}$ initiates the process by sending an authentication request (challenge) to the IoT device D_k. Upon receiving the challenge, D_k randomly selects $\rho_k \xleftarrow{\$} \mathbb{F}^*_{q_a}$, then computes $R_k = \rho_k \cdot P_g$ and $K = \rho_k \cdot \mathsf{PK}_{\mathcal{S}}$. It masks K as $M = hID_{D_k} \oplus K$ and generates the authentication tag $\mathsf{Auth}_k = \mathsf{H}(ID_{D_k} \parallel 2 \parallel K)$. Finally, it sends $(R_k, \mathsf{Auth}_k, M)$ to $\mathcal{S}$.
- Device authentication by server: The server computes $K = \mathsf{SK}_{\mathcal{S}} \cdot R_k$ and derives $hID'_{D_k} = M \oplus K$. It then searches its database for a matching ID'_{D_k} using hID'_{D_k}. If no match is found, the session is terminated after a delay to limit information leakage. If a match exists, the server calculates $\mathsf{Auth}'_k = \mathsf{H}(ID'_{D_k} \parallel 2 \parallel K)$ and compares it with Auth_k. If the values match, D_k is authenticated; otherwise, the session is aborted. Upon successful authentication, the server generates $\mathsf{Auth}_{\mathcal{S}} = \mathsf{H}(ID_{D_k} \parallel 1 \parallel K)$ and sends it to the device.

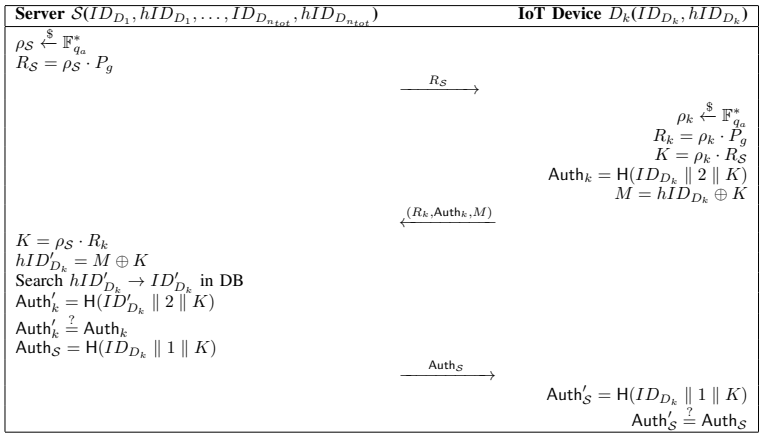

Fig. 3. Authentication phase.

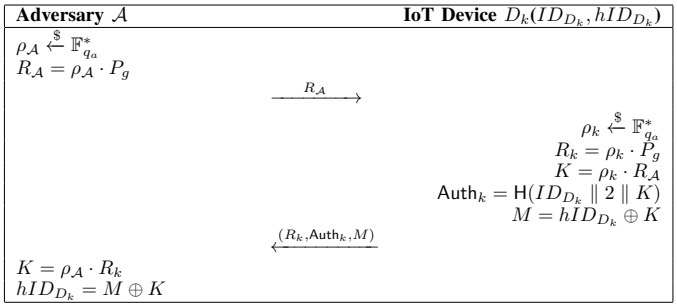

Fig. 4. Recovery of hID_{D_k}.

- Server authentication by device: After receiving $\mathsf{Auth}_{\mathcal{S}}$, the device computes $\mathsf{Auth}'_{\mathcal{S}} = \mathsf{H}(ID_{D_k} \| 1 \| K)$ and verifies it against the received value. If they match, $\mathcal{S}$ is authenticated, and mutual authentication is established; otherwise, the session is terminated.

At the conclusion of this phase, both D_k and $\mathcal{S}$ share the session key K, computed as:

$$K = \rho_k \cdot \mathsf{PK}_{\mathcal{S}} = \mathsf{SK}_{\mathcal{S}} \cdot R_k = \mathsf{SK}_{\mathcal{S}} \cdot \rho_k \cdot P_g.$$

This shared key is subsequently used to ensure integrity, confidentiality, and privacy in all subsequent communications.

VIII. Security Analysis

A. Informal Analysis

1) Mutual authentication:

a) Device authentication: An adversary is unable to generate a valid message $(R_k, \mathsf{Auth}_k, M)$ on behalf of device D_k, as they lack the necessary values hID_{D_k}, ID_{D_k}, and ρ_k. These parameters are essential for computing Auth_k and M, and can only be derived by the legitimate server $\mathcal{S}$ and device D_k.

b) Server authentication: The adversary cannot forge the legitimate server message $\mathsf{Auth}_{\mathcal{S}}$, since it requires knowledge of the identifier ID_{D_k}, which is not disclosed.

TABLE II. Results of the Verification by AVISPA

back-end	Result
CL-AtSe	SAFE
OFMC	SAFE
SATMC	INCONCLUSIVE
TA4SP	INCONCLUSIVE

2) Integrity and confidentiality: Upon completion of the protocol, the server $\mathcal{S}$ and device D_k establish a shared session key K. This key is jointly derived from $\mathsf{SK}_{\mathcal{S}}$ (generated by $\mathcal{S}$) and ρ_k (generated by D_k) for each session, thereby ensuring both message integrity and confidentiality.

3) Availability:

a) Resistance to Man-in-the-Middle (MitM) attacks: The adversary cannot obtain ID_{D_k}, hID_{D_k}, or the session key K. In contrast, the protocol of Fatma et al. [6] is susceptible to MitM attacks, as the session key K can be recovered by an adversary.

B. Formal Analysis

In this section, we conduct a formal security assessment of the proposed scheme using two widely recognized automated verification tools: AVISPA [31] and ProVerif [32].

1) HLPSL Code of the proposed protocol: The HLPSL specification of our protocol is provided in Fig. 8 and Fig. 9. For a more comprehensive explanation, including extended formal verification procedures and additional security analysis, readers are referred to our related conference publication [33].

2) AVISPA verification results: Table II summarizes the outcomes obtained after executing our HLPSL-coded protocol in the AVISPA environment. The results indicate that both the CL-AtSe and OFMC back-ends report the status SAFE, confirming the absence of detected attacks. For the SATMC and TA4SP back-ends, the outcome is INCONCLUSIVE due to unsupported operations, which implies that these modules are unable to process certain steps of the protocol rather than indicating any security weakness.

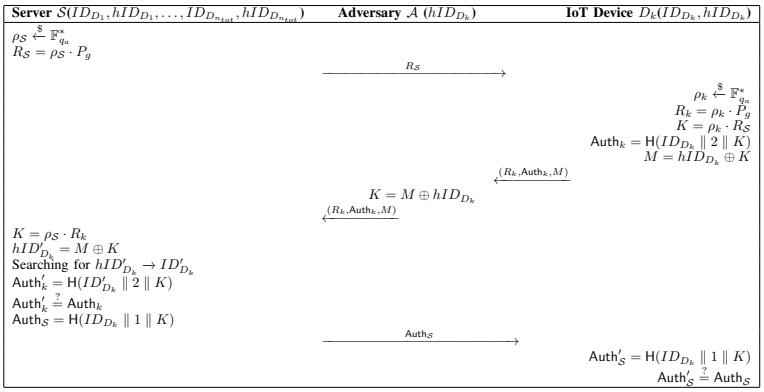

Fig. 5. Recovery of the key K.

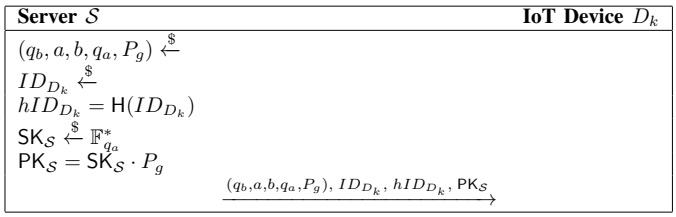

Fig. 6. ECC-based configuration phase.

C. Specification of Our Protocol using the Pi-Calculus Script in ProVerif

In this section, we will present the code of our protocol using Pi-Calculus. Fig. 10 and Fig. 11 presents the server role and smart object role.

1) The results of the verification by Proverif: Fig. 12 represents the results of our contributions using the ProVerif tool. It shows that mutual authentication is achieved, and the confidentiality of variables idt, hidt, ri, and rs is preserved.

IX. Performance Evaluation

This section presents an evaluation of the proposed scheme with respect to communication overhead, and computational effort. For the analysis, we consider a deployment consisting of n_{tot} IoT devices. We assume that the size of a scalar value is λ bits, which implies that the size of a point on the selected elliptic curve $\mathcal{E}$ is 2λ bits. Accordingly, the size of the system parameters (q_b, a, b, q_a, P_g) is taken to be 6λ bits in total. Furthermore, both the device identifier ID_{D_k} and its corresponding hashed representation hID_{D_k} are assumed to be scalar values, each occupying λ bits.

T_{PM} and T_{PA} represent the execution time required to perform elliptic curve point multiplication and point addition operations, respectively.

A. Computation Cost

In this section, we assess and compare the computational effort required by both the server (see Table IV) and the smart objects (see Table III) during the authentication process. The performance of our proposed protocol is evaluated against several existing schemes, including those presented by Liao et al. [29], Dinarvand et al. [28], Jin et al. [34], Alamr et al. [26], Zhao [30], and Fatma et al. [6].

TABLE III. Evaluation of Computation Costs for Smart Devices

approach	Computational Overheads
Zhao et al. [30]	$3 * T_{\text{PA}} + 5 * T_{\text{PM}}$
Dinarvand et al. [28]	$2 * T_{\text{PA}} + 3 * T_{\text{PM}}$
Jin et al. [34]	$1 * T_{\text{PA}} + 4 * T_{\text{PM}}$
Alamr et al. [26]	$1 * T_{\text{PA}} + 4 * T_{\text{PM}}$
Liao et al. [29]	$3 * T_{\text{PA}} + 5 * T_{\text{PM}}$
Fatma et al. [6]	$2 * T_{\text{PM}}$
Our approach	$2 * T_{\text{PM}}$

TABLE IV. Evaluation of Computation Costs for Server

approach	Computational Overheads
Zhao et al. [30]	$3 * T_{\text{PA}} + 5 * T_{\text{PM}}$
Dinarvand et al. [28]	$2 * T_{\text{PA}} + 3 * T_{\text{PM}}$
Jin et al. [34]	$2 * T_{\text{PM}}$
Alamr et al. [26]	$1 * T_{\text{PA}} + 5 * T_{\text{PM}}$
Liao et al. [29]	$3 * T_{\text{PA}} + 5 * T_{\text{PM}}$
Fatma et al. [6]	$2 * T_{\text{PM}}$
Our approach	$1 * T_{\text{PM}}$

As shown in Table IV, our proposed improvement achieves lower server computation costs than the compared protocols, while maintaining a database search complexity of $O(1)$, similar to the approach of Fatma et al. [6].

B. Transmission Overhead

The transmission Overhead is assessed by determining the bit length of all messages exchanged during the authentication phase, along with counting the total number of message transmissions.

In Table V, our enhancement is the best protocol in terms of computation cost at the server level.

X. Practical Analysis

To evaluate the feasibility and performance of the proposed ECC-based mutual authentication protocol, a real-world imple-

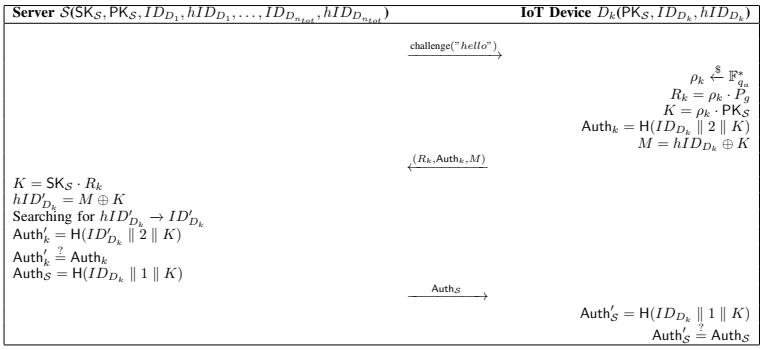

Fig. 7. ECC-based authentication phase.

Server role

$role\ serverRole(A, B : agent, Q, Idx, HashX, Txt : text, Msgs : message, F, HashFunc : hash\_func, SEND, RECEIVE : channel(dy))$
$played\_byA$
$def =$
  $local$
    $Status : nat, Rnd, Key : text,$
    $Rq, Msg, AuthA, AuthB : message$
    $const sec\_rnd, sec\_idx, sec\_hashx, auth\_b, auth\_a : protocol\_id$
  $init$
    $Status := 0$
  $transition$
    $1.Status = 0 \ \wedge \ RECEIVE(start) \Rightarrow Status' := 2 \ \wedge Rnd' := new()$
      $\wedge SEND(Rnd)$
    $2.Status = 2 \ \wedge \ RECEIVE(Rq', AuthB', Msg')$
      $\wedge Key' = HashFunc(Txt, Rq')$
      $\wedge HashX = xor(Msg', Key')$
      $\wedge AuthB = F(Idx'.2.Key') \Rightarrow Status' := 4 \ \wedge AuthA' := F(Idx.1.Key')$
        $\wedge SEND(AuthA')$
        $\wedge secret(Idx, sec\_idx, A)$
        $\wedge secret(HashX, sec\_hashx, A)$
        $\wedge request(A, B, auth\_b, AuthB')$
        $\wedge witness(A, B, auth\_a, AuthA')$
$end\ role$

Fig. 8. Server role of HLPSL.

Object role

$role\ objectRole(B, A : agent, Q, Ident, HashIdent : text, MsgList : message, F, HashFunc : hash\_func, SEND, RECEIVE : channel(dy))$
$played\_byB$
$def =$
  $local$
    $Status : nat, Random, Key : text,$
    $ReqIdent, RespIdent, Msg, AuthA, AuthB : message$
    $const sec\_random, sec\_ident, sec\_hashident, auth\_a, auth\_b : protocol\_id$
  $init$
    $Status := 1$
  $transition$
    $1.Status = 1 \ \wedge \ RECEIVE(RespIdent') \Rightarrow Status' := 3 \ \wedge Random' := new()$
      $\wedge ReqIdent' := HashFunc(Random', Q)$
      $\wedge Key' := HashFunc(Random', MsgList)$
      $\wedge AuthB' := F(Ident.2.Key')$
      $\wedge Msg' := xor(HashIdent, Key')$
      $\wedge SEND(ReqIdent', AuthB', Msg')$
      $\wedge secret(Ident, sec\_ident, B)$
      $\wedge secret(HashIdent, sec\_hashident, B)$
      $\wedge secret(Random', sec\_random, B)$
      $\wedge witness(B, A, auth\_a, AuthB')$
    $2.Status = 3 \ \wedge \ RECEIVE(AuthA')$
      $\wedge AuthA' = F(Ident.1.Key) \Rightarrow Status' := 5 \ \wedge request(B, A, auth\_b, AuthA')$
$end\ role$

Fig. 9. Role of object B.

mentation was conducted using two heterogeneous physical devices: a Raspberry Pi 3 Model B+ representing the IoT object, and a standard personal computer equipped with an Intel Core i5 (5th generation) processor functioning as the central server. The Raspberry Pi, operating under a Linux-based Raspbian OS, was chosen to simulate the constraints typical of embedded IoT devices in terms of memory, CPU capacity, and energy consumption. The server ran Ubuntu 20.04 and was responsible for handling the more computationally intensive cryptographic verifications and key exchange processes. The protocol logic was fully developed in Python 3 using the ecdsa cryptographic library, which provides implementations of elliptic curve operations and ECDSA signatures. Experiments were conducted using a range of elliptic curves to evaluate their impact on the protocol's performance. The tested curves include: NIST192p, NIST224p, NIST256p, SECP256k1, BRAINPOOLP160r1, BRAINPOOLP192r1, BRAINPOOLP256r1, and Ed25519. These curves were used to measure the execution time of the authentication phase for each configuration, and the results obtained are presented in Fig. 13.

Server Role

```
(* Role of the Server S *)
let SERVERS =
    new rs : bitstring;
    let RS = mecc(rs, p) in
    out(ch, RS);
      in(ch, (Ri : bitstring, authi : bitstring, hidt_recv : bitstring));
    let K = mecc(Ri, rs) in
    if authi = h(concat(idt, concat(v2, K))) then
        let AUTHS = h(concat(idt, concat(v1, K))) in
        event beginS (AUTHS);
        out(ch, AUTHS);
        event endS (AUTHS).
```

Fig. 10. Server role.

Smart Object Role

```
(* Role of the Smart Object I *)
let SOI =
    in (ch, RS : bitstring);
    new ri : bitstring;
    let RI = mecc(ri, p) in
    let K = mecc(ri, RS) in
    let AUTHI = h(concat(idt, concat(v2, K))) in
    event beginT (AUTHI);
    out (ch, (RI, AUTHI, hidt));
    in (ch, auths : bitstring);
    if auths = h(concat(idt, concat(v1, K))) then
        event endT (AUTHI).
```

Fig. 11. Smart object role.

```
------------------------------------------------------------
Verification summary:

Query not attacker(idt[]) is true.

Query not attacker(ri[RS_1 = v,!1 = v_1]) is true.

Query not attacker(rs[!1 = v]) is true.

Query inj-event(beginS(x)) ==> inj-event(beginT(x)) is true.

Query inj-event(endT(x)) ==> inj-event(endS(x)) is true.

Query inj-event(endS(y)) ==> inj-event(beginS(x)) && inj-event(beginT(x)) is true.

------------------------------------------------------------
```

Fig. 12. Results of the verification by Proverif.

TABLE V. COMPARISON OF TRANSMISSION OVERHEAD

approach	S → O	O → S	Total (bits)	Number of Messages
Alamr et al. [26]	$4 * \lambda$	$6 * \lambda$	$10 * \lambda$	3
Dinarvand et al. [28]	$5 * \lambda$	$4 * \lambda$	$9 * \lambda$	4
Jin et al. [34]	$4 * \lambda$	$3 * \lambda$	$7 * \lambda$	3
Zhao et al. [30]	$4 * \lambda$	$4 * \lambda$	$8 * \lambda$	3
Liao et al. [29]	$4 * \lambda$	$4 * \lambda$	$8 * \lambda$	3
Fatma et al. [6]	$4 * \lambda$	$3 * \lambda$	$7 * \lambda$	3
Our proposed	$4 * \lambda$	$2 * \lambda$	$6 * \lambda$	3

This implementation setup allowed for real-time testing of the protocol under realistic hardware constraints and provided concrete insights into its computational efficiency and practicality for IoT deployments.

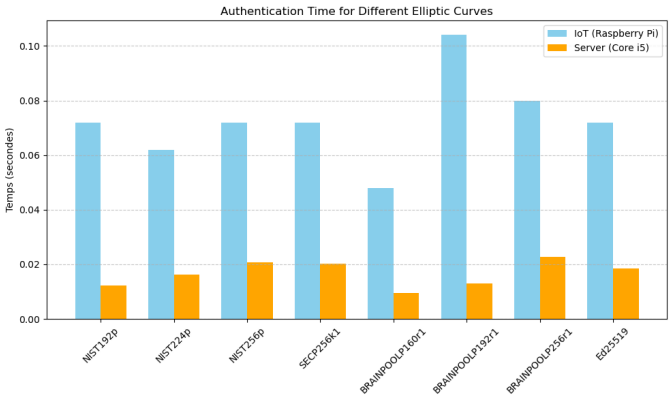

Fig. 13. Authentication time for different elliptic curves.

A. Computational Cost of Our Protocol

The average execution time of our authentication protocol, measured across the full exchange phase between the IoT device (Raspberry Pi) and the server (Core i5), is approximately 0.089348 seconds, consisting of 0.072703 seconds on the IoT side and 0.016645 seconds on the server side. These values were obtained by computing the mean execution time across a range of standardized elliptic curves.

This implementation demonstrated that our protocol successfully reduced the execution time compared to the protocol proposed by Fatma et al., particularly during the authentication phase, as illustrated in Fig. 14 . This noticeable reduction in processing time highlights the effectiveness of our approach, which combines computational efficiency with robust security. As a result, the proposed protocol is well suited for real-time execution on resource-constrained IoT platforms, while ensuring secure and mutual authentication.

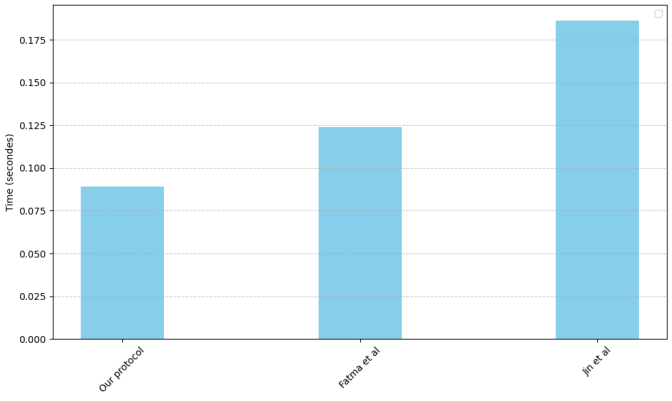

Fig. 14. Comparison of computation costs.

XI. CONCLUSION

This research presents a comprehensive study of an efficient protocol originally proposed by Fatma et al. Our security

analysis revealed several critical vulnerabilities that could compromise the robustness of the protocol in real-world IoT deployments. In response to these weaknesses, we proposed a reinforced version of the protocol, focusing on both improving security guarantees and reducing computational and communication overhead. To validate the proposed enhancements, we not only employed formal verification tools, such as AVISPA and ProVerif but also carried out a practical implementation using heterogeneous devices—specifically, a Raspberry Pi 3 Model B+ and a standard PC—to simulate constrained IoT environments. This real-world experimentation allowed us to analyze execution times and resource consumption across a variety of elliptic curves. The results, drawn from both simulation and implementation, clearly demonstrate that the enhanced protocol significantly improves upon the original in terms of resistance to known attacks and performance efficiency. These outcomes confirm the protocol's relevance and feasibility for secure authentication in IoT systems and highlight the value of combining formal analysis with empirical validation to design resilient and practical security solutions.

REFERENCES

[1] H. Zougagh, N. Idboufker, and Y. Saadi, "Trust-based intrusion detection for multi-path olsr protocol," in *Proceedings of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2020)*, vol. 12. Springer International Publishing, 2021, pp. 690–705.

[2] H. Zougagh, N. Idboufker, Y. El Mourabit, Y. Saadi, and S. Elouaham, "Avoiding wormhole attack in manet using an extending network knowledge," in *Innovations in Bio-Inspired Computing and Applications: Proceedings of the 11th International Conference on Innovations in Bio-Inspired Computing and Applications (IBICA 2020) held during December 16-18, 2020*, vol. 11. Springer International Publishing, 2021, pp. 217–230.

[3] Z. Elhadari, H. Zougagh, N. Idboufker, and M. Ech-chebaby, "Survey on the adoption of blockchain technology in internet of things environments: Techniques, challenges and future research directions," *IAENG International Journal of Computer Science*, vol. 52, no. 1, pp. 59–89, 2025.

[4] M. Naji and H. Zougagh, "Deep learning models for cybersecurity in iot networks," in *International Conference on Business Intelligence*. Springer Nature Switzerland, July 2023, pp. 29–43.

[5] H. Zougagh, N. Idboufker, R. Zoubairi, and R. El Ayachi, "Prevention of black hole attacks on mobile ad hoc networks through intrusion detection systems," *International Journal of Business Data Communications and Networking (IJBDCN)*, vol. 15, no. 2, pp. 73–91, 2019.

[6] F. Merabet, A. Cherif, M. Belkadi, O. Blazy, E. Conchon, and D. Sauveron, "New efficient m2c and m2m mutual authentication protocols for IoT-based healthcare applications," vol. 13, no. 2, pp. 439–474, 03 2020. [Online]. Available: https://doi.org/10.1007/s12083-019-00782-8

[7] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[8] S. Pote, V. Sule, and B. K. Lande, "Arithmetic of koblitz curve *Secp256k1* used in bitcoin cryptocurrency based on one variable polynomial division," 04 2019. [Online]. Available: https://papers.ssrn.com/abstract=3367674

[9] J. H. Silverman and J. Suzuki, "Elliptic curve discrete logarithms and the index calculus," in *Advances in Cryptology — ASIACRYPT'98*, K. Ohta and D. Pei, Eds. Springer Berlin Heidelberg, 1998, pp. 110–125.

[10] P. Gaudry, "Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem," vol. 44, no. 12, pp. 1690–1702, 12 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S074771710800182X

[11] A. Srivastava and A. Kumar, "A review on authentication protocol and ecc in iot," in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2021, pp. 312–319.

[12] B. Preneel, "Cryptographic hash functions," vol. 5, no. 4, pp. 431–448, 1994, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.4460050406. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4460050406

[13] X. Wang and H. Yu, "How to break MD5 and other hash functions," in *Advances in Cryptology – EUROCRYPT 2005*, R. Cramer, Ed. Springer Berlin Heidelberg, 2005, pp. 19–35.

[14] N. Kheshaifaty and A. Gutub, "Preventing multiple accessing attacks via efficient integration of captcha crypto hash functions," vol. 20, pp. 16–28, 09 2020.

[15] Rohit, S. Kamra, M. Sharma, and A. Leekha, "Secure hashing algorithms and their comparison," in *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2019, pp. 788–792.

[16] A. A. Yaseen, K. Patel, A. A. Yassin, A. J. Aldarwish, and H. A. Hussein, "Secure electronic healthcare record using robust authentication scheme," *IAENG International Journal of Computer Science*, vol. 50, no. 2, pp. 468–476, 2023.

[17] S. Boonkrong, "Security analysis and improvement of a multi-factor biometric-based remote authentication scheme," *IAENG International Journal of Computer Science*, vol. 46, no. 4, pp. 713–724, 2019.

[18] Y. Park, K. Park, K. Lee, H. Song, and Y. Park, "Security analysis and enhancements of an improved multi-factor biometric authentication scheme," *International Journal of Distributed Sensor Networks*, vol. 13, no. 8, p. 1550147717724308, 2017. [Online]. Available: https://doi.org/10.1177/1550147717724308

[19] L. Zhang, Y. Zhang, S. Tang, and H. Luo, "Privacy protection for e-health systems by means of dynamic authentication and three-factor key agreement," *IEEE Transactions on Industrial Electronics*, vol. 65, pp. 2795–2805, 2018.

[20] Q. Jiang, J. Ma, C. Yang, X. Ma, J. Shen, and S. A. Chaudhry, "Efficient end-to-end authentication protocol for wearable health monitoring systems," vol. 63, pp. 182–195, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790617305128

[21] R. Amin, S. H. Islam, G. P. Biswas, M. K. Khan, and N. Kumar, "A robust and anonymous patient monitoring system using wireless medical sensor networks," vol. 80, pp. 483–495, 03 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X16301509

[22] X. Li, J. Peng, S. Kumari, F. Wu, M. Karuppiah, and K.-K. Raymond Choo, "An enhanced 1-round authentication protocol for wireless body area networks with user anonymity," vol. 61, pp. 238–249, 07 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790617303099

[23] J. Liu, L. Zhang, and R. Sun, "1-RAAP: An efficient 1-round anonymous authentication protocol for wireless body area networks," vol. 16, no. 5, p. 728, 05 2016, number: 5 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1424-8220/16/5/728

[24] N. Yessad, S. Bouchelaghem, F.-S. Ouada, and M. Omar, "Secure and reliable patient body motion based authentication approach for medical body area networks," vol. 42, pp. 351–370, 12 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574119217303048

[25] C. Jin, C. Xu, X. Zhang, and F. Li, "A secure ECC-based RFID mutual authentication protocol to enhance patient medication safety," vol. 40, no. 1, p. 12, 10 2015. [Online]. Available: https://doi.org/10.1007/s10916-015-0362-8

[26] A. A. Alamr, F. Kausar, J. Kim, and C. Seo, "A secure ECC-based RFID mutual authentication protocol for internet of things," vol. 74, no. 9, pp. 4281–4294, 09 2018, company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 9 Publisher: Springer US. [Online]. Available: https://link.springer.com/article/10.1007/s11227-016-1861-1

[27] N. Mehibel and M. Hamadouche, "A new approach of elliptic curve diffie-hellman key exchange," in *2017 5th International Conference on Electrical Engineering - Boumerdes (ICEE-B)*, 2017, pp. 1–6.

[28] N. Dinarvand and H. Barati, "An efficient and secure RFID authentication protocol using elliptic curve cryptography," vol. 25, no. 1, pp. 415–428, 01 2019. [Online]. Available: https://doi.org/10.1007/s11276-017-1565-3

[29] Y.-P. Liao and C.-M. Hsiao, "A secure ECC-based RFID authentication scheme using hybrid protocols," in *Advances in Intelligent Systems and Applications - Volume 2*, ser. Smart Innovation, Systems and Technologies, J.-S. Pan, C.-N. Yang, and C.-C. Lin, Eds. Springer, 2013, pp. 1–13.

[30] Z. Zhao *et al.*, "A secure rfid authentication protocol for healthcare environments using elliptic curve cryptosystem," *Journal of Medical Systems*, vol. 38, no. 5, p. 46, 2014. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/24756871/

[31] L. Viganò, "Automated security protocol analysis with the AVISPA tool," vol. 155, pp. 61–86, 05 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1571066106001897

[32] B. Blanchet, "Automatic verification of security protocols in the symbolic model: The verifier ProVerif," in *Foundations of Security Analysis and Design VII: FOSAD 2012/2013 Tutorial Lectures*, ser. Lecture Notes in Computer Science, A. Aldini, J. Lopez, and F. Martinelli, Eds. Springer International Publishing, 2014, pp. 54–87. [Online]. Available: https://doi.org/10.1007/978-3-319-10082-1_3

[33] M. Ech-chebaby, H. Zougagh, H. Garmani, Z. El-hadari, and N. Idboufker, "Enhanced ecc-based authentication protocol for iot devices m2c," in *Proceedings of the 4th International Conference on Advances in Communication Technology and Computer Engineering (ICACTCE'24)*, C. Iwendi, Z. Boulouard, and N. Kryvinska, Eds. Cham: Springer Nature Switzerland, 2025, pp. 154–169.

[34] C. Jin, C. Xu, X. Zhang, and F. Li, "A secure ECC-based RFID mutual authentication protocol to enhance patient medication safety," vol. 40, no. 1, p. 12, 01 2016. [Online]. Available: https://doi.org/10.1007/s10916-015-0362-8