

# LAPWN: A Lightweight User–Server Authentication Protocol for Wireless Networks

Sajjad Alizadeh<sup>1,c</sup>, Reza Hooshmand<sup>2</sup>

<sup>1</sup> Department of Electrical Engineering, Yazd University, Yazd, Iran, Postal Code: 89195-741

<sup>2</sup> Department of Electrical Engineering, Shahid Sattari University of Aeronautical Science and Technology, Tehran, Iran, Postal Code: 13846-63113

<sup>c</sup> Corresponding Author: Sajjad Alizadeh, Email: [sajjad.alizadeh.academic@gmail.com](mailto:sajjad.alizadeh.academic@gmail.com)

## ARTICLE INFO

### Keywords:

Lightweight  
User–Server Authentication  
Wireless Networks  
Resource-Constrained Environments  
Scyther  
ProVerif  
Formal Verification  
Python

## Abstract

The Internet of Things (IoT) is composed of interconnected devices that exchange data over a network, enabling applications in healthcare, transportation, and smart environments. As IoT ecosystems expand, ensuring security and privacy remains a critical challenge. Many IoT devices rely on wireless networks for data transmission, making them vulnerable to eavesdropping, tracking, and tampering. This highlights the need for robust authentication mechanisms. To address these concerns, numerous authentication protocols have been proposed. However, many fail to ensure adequate security against both passive and active attacks. In this research, we introduce LAPWN, a lightweight protocol for user–server communication, specifically designed for constrained environments, ensuring a balance between security and efficiency. The proposed protocol is implemented as a fully functional Python application, demonstrating its practical usability and evaluating its efficiency in real-world scenarios. To validate its security, we perform both informal and formal analyses, utilizing Scyther, ProVerif, and the Real-or-Random (RoR) model. The results confirm that LAPWN provides a secure, lightweight, and efficient authentication solution with low computational and communication overhead. Furthermore, performance evaluations show that it surpasses existing authentication protocols, making it a highly effective solution for secure user–server interactions in constrained environments.

## 1. Introduction

The Internet of Things (IoT) has been widely adopted in smart cities, transportation, home automation, and particularly healthcare, where it plays a crucial role in enabling intelligent services. In modern healthcare infrastructures, IoT devices—often connected via wireless networks—facilitate secure real-time communication between users (e.g., patients or medical staff) and remote servers (e.g., hospital databases or telemedicine platforms). This interaction enables remote access to medical data, patient monitoring, and efficient healthcare service management [Sadhukhan et al. \(2021\)](#).

However, since these interactions often occur over open or semi-secure wireless channels, they pose significant security risks. Robust user–server authentication mechanisms are essential to protect patient privacy and ensure the integrity of transmitted data. Without adequate authentication, adversaries can impersonate users or servers, manipulate health records, or intercept sensitive medical information, leading to severe security and privacy breaches [Chen et al. \(2019\)](#).


A well-designed user–server authentication protocol must address several key security challenges. It should enable mutual authentication, ensuring that both the user and the server verify each other's legitimacy before exchanging sensitive information. Additionally, the protocol must incorporate anonymity and untraceability to prevent adversaries from linking authentication attempts to specific users. Secure session key establishment is also crucial for ensuring confidentiality and data integrity during transmission. Fur-

thermore, authentication schemes must be resilient against various attack vectors, including impersonation, replay, and man-in-the-middle attacks [Suganthi et al. \(2020\)](#), especially in wireless and resource-constrained IoT environments.

Numerous authentication mechanisms have been proposed for IoT-based user–server communication in healthcare scenarios. However, many fail to efficiently establish user–server authentication while minimizing computational and communication overhead. Existing solutions often introduce excessive processing costs—making them unsuitable for low-power or bandwidth-limited devices—or lack essential security properties such as anonymity, forward secrecy, and resistance to known cyber threats.

To address these limitations, this research introduces LAPWN, a lightweight and efficient user–server authentication protocol for wireless IoT systems. The proposed scheme ensures secure communication while reducing computational and communication overhead. It facilitates a challenge–response procedure that establishes mutual trust between the user and the server. Additionally, it enables secure session key agreement, ensuring that transmitted medical (or similarly sensitive) data remains confidential and tamper-proof while preventing adversaries from linking authentication requests to specific users. The protocol also supports a secure password update phase, allowing legitimate users to modify their credentials without compromising security.

To ensure the security of our proposed authentication protocol, we leverage the Real-or-Random (RoR) proof model, which provides a rigorous framework for assessing the protocol's robustness against cyber threats. Furthermore, we perform comprehensive formal security analyses using Scyther and ProVerif, demonstrating the protocol's resilience against

 [sajjad.alizadeh.academic@gmail.com](mailto:sajjad.alizadeh.academic@gmail.com) (S. Alizadeh<sup>1,c</sup>);

[rhooshmand50@yahoo.com](mailto:rhooshmand50@yahoo.com) (R. Hooshmand<sup>2</sup>)

ORCID(s):

various authentication-based attacks. Finally, a Python-based implementation is developed to assess its real-world performance in IoT-driven healthcare scenarios. The results show that LAPWN maintains a strong balance between security, computational efficiency, and low communication cost, making it a highly effective solution for ensuring secure authentication in resource-constrained, wireless IoT environments. By addressing the shortcomings of existing solutions, this protocol contributes to the advancement of secure authentication mechanisms in modern IoT-based healthcare applications.

### 1.1. Motivation and Contributions

In modern smart environments such as healthcare systems, secure data transmission is crucial to maintaining the integrity and confidentiality of sensitive information. User authentication plays a fundamental role in ensuring that only legitimate users can access protected resources. In IoT-driven medical applications, patient data is frequently exchanged between users and healthcare servers—often over wireless or public networks—making it susceptible to various security threats. To mitigate these risks, an efficient and lightweight authentication mechanism is essential. This research introduces a novel user–server authentication protocol designed specifically for securing IoT-based healthcare applications, though it can also be applied to other resource-constrained or wireless environments.

1. We propose a lightweight authentication scheme for IoT-based healthcare environments, ensuring strong security guarantees with minimal computational and communication overhead. This makes it highly suitable for resource-constrained systems.
2. We conduct a comprehensive security evaluation of the proposed scheme, utilizing both theoretical and automated security analyses. Formal validation is performed using tools such as Scyther and ProVerif, while additional assessments are conducted based on the Real-or-Random (RoR) model. These analyses confirm that the protocol effectively mitigates various security threats, including identity spoofing, replay attacks, and unauthorized access attempts.
3. We develop a practical implementation of the authentication protocol using Python 3.9, demonstrating its real-world applicability. This implementation serves as a proof of concept, showcasing its feasibility in IoT-driven healthcare systems and other environments with similar constraints.
4. We perform a detailed performance evaluation, comparing the proposed protocol against existing state-of-the-art authentication schemes. The results indicate that our protocol maintains an optimal balance between security and efficiency, making it a viable solution for lightweight authentication in modern IoT-based applications.

By addressing existing authentication challenges in IoT-driven healthcare applications, this research contributes to the de-

velopment of secure, efficient, and scalable authentication mechanisms for modern smart environments.

### 1.2. Paper Organization

The organization of this paper is as follows: Section 2 provides an overview of related research, discussing existing authentication approaches and their limitations. Section 3 outlines the architecture of the health monitoring system, highlighting its security challenges and essential requirements. The proposed user–server authentication protocol, *LAPWN*, is introduced in Section 4, detailing its design principles and security objectives.

A comprehensive security evaluation of *LAPWN* is conducted in Section 5, encompassing both theoretical analysis and formal verification using automated tools. Section 6 presents the implementation details of the proposed scheme, including its development in Python 3.9 and its applicability to real-world healthcare IoT environments. In Section 7, the performance of *LAPWN* is analyzed and compared with existing authentication mechanisms to assess its efficiency and security improvements. Finally, Section 8 wraps up the paper by examining the protocol’s limitations and suggesting possible avenues for future research.

## 2. Related Work

In recent years, numerous authentication protocols have been introduced to enhance security in IoT-based healthcare environments. These schemes primarily focus on lightweight cryptographic mechanisms, elliptic curve cryptography (ECC), and privacy-preserving techniques for secure authentication. Table 1 provides a concise overview of key contributions in this field. However, despite ongoing advancements, many existing authentication mechanisms have been found vulnerable to security breaches, including identity impersonation, replay attacks, session key disclosure, and privacy violations. Several authentication schemes have been proposed for healthcare IoT applications, yet many fail to provide robust security guarantees. For instance, Kumar et al. (2019) introduced a cloud-assisted authentication scheme for telemedicine information systems (TMIS), claiming resistance against privacy and security threats. However, subsequent research by Kumar et al. (2019) identified weaknesses, including susceptibility to impersonation attacks and privacy breaches. Similarly, Zheng et al. (2018) presented an authentication model applicable to smart campus networks, incorporating TMIS. However, later research by Safkhani and Vasilakos (2019) demonstrated that their scheme was vulnerable to impersonation and replay attacks.

The security of home automation and IoT-based smart grid authentication has also been investigated. For example, Xiang and Zheng (2020) proposed a situation-aware authentication protocol for device authentication within smart home networks and smart grids. However, Oh et al. (2021) revealed multiple vulnerabilities in this scheme, including susceptibility to stolen device attacks, impersonation threats, and session key leakage, making it unsuitable for practical deployment. Additionally, Shuai et al. (2019) introduced

an ECC-based anonymous authentication method for smart home environments. While this approach aimed to preserve user privacy, further evaluation revealed security gaps, including the potential for adversaries to track user identities. Wearable medical devices and healthcare authentication schemes have also been widely studied. Gupta et al. (2019) introduced a lightweight authentication scheme for wearable sensor devices, but it was later demonstrated by Hajian et al. (2020) that the protocol suffered from critical vulnerabilities, including privileged insider threats, de-synchronization issues, and compromised sensing devices. Similarly, Xu et al. (2019) designed an authentication protocol claimed to be resilient against various attacks. However, Alzahrani et al. (2021) later proved that the scheme was vulnerable to key compromise, replay, and impersonation attacks.

In the field of ultra-lightweight cryptography, Wang et al. (2022) and Chander and Gopalakrishnan (2022) proposed highly efficient authentication mechanisms. However, Servati et al. (2022) demonstrated that the protocol introduced by Wang et al. (2022) was susceptible to secret disclosure and de-synchronization attacks. Additionally, Aghili et al. (2019a) suggested a SecLAP protocol using lightweight rotation-based operations, but Safkhani et al. (2020) later identified traceability and secret value leakage vulnerabilities within this model.

ECC-based authentication protocols have been extensively examined for IoT security. Gabsi et al. (2021) proposed an ECC-based authentication model for IoT applications, which was initially deemed secure. However, subsequent research by Arslan and Bingöl (2022) uncovered several vulnerabilities, including traceability issues, forward and backward secrecy contradictions, and tag anonymity violations.

Another notable ECC-based authentication scheme was proposed by Huang (2024), focusing on three-factor authentication for wireless sensor networks (WSNs). This method aimed to enhance both security and computational efficiency in resource-limited environments. Additionally, Kumar et al. (2024) introduced the 2F-MASK-VSS, a two-factor authentication system designed to improve security in video surveillance systems. Meanwhile, Rani and Tripathi (2024) developed a blockchain-based authentication mechanism for secure health data sharing across hospitals, leveraging blockchain's decentralized properties for better access control and data integrity.

Advancements in key management and authentication efficiency for IoT-based WSNs were also explored by Chatterjee et al. (2023), who presented an ECC-based model to enhance security and computational efficiency. In parallel, Wang et al. (2024) proposed a dual-layered authentication framework integrating ECC with blockchain for secure IoT device identification, ensuring both scalability and security resilience. In the healthcare sector, Khan et al. (2023b) introduced a robust authentication scheme for e-healthcare systems, aiming to strengthen data protection and privacy in patient health record exchanges.

Several lightweight and blockchain-based authentication frameworks have also been proposed. For instance, Jegadeesan et al. (2022) introduced a lightweight authentication scheme optimized for resource-constrained healthcare systems. Similarly, Jia et al. (2022) developed a blockchain-enhanced authentication model for healthcare applications, addressing privacy and security concerns. However, both protocols exhibited security limitations, as Jegadeesan et al. (2022) failed to prevent user traceability, while Jia et al. (2022) was shown to allow extraction of critical security parameters.

Recent authentication frameworks have focused on addressing these shortcomings by integrating enhanced cryptographic techniques. Wang and Liu (2022) introduced the RC2PAS protocol, designed to improve authentication security in resource-constrained networks. Meanwhile, Pu et al. (2022) developed a lightweight authentication model specifically for wireless body area networks, ensuring high security while maintaining computational efficiency. Additionally, Shariq et al. (2021) and Khan et al. (2023a) proposed authentication solutions tailored for RFID and constrained IoT systems, yet Hosseinzadeh et al. (2024) later demonstrated their susceptibility to secret disclosure vulnerabilities. Furthermore, Hosseinzadeh et al. (2024) introduced an improved authentication scheme that effectively mitigates these security risks.

Authentication in medical IoT environments remains a significant research focus. A recently proposed authentication protocol introduced by Servati et al. (2025) enhances security while maintaining computational efficiency. Through extensive formal security analyses using the Real-or-Random (RoR) model, Scyther, and ProVerif, the study demonstrated that the protocol mitigates various security threats while ensuring a lightweight structure suitable for healthcare applications.

In summary, while numerous authentication mechanisms have been proposed to secure IoT-driven healthcare environments, many remain susceptible to critical vulnerabilities. Emerging cryptographic techniques, including ECC, blockchain integration, and multi-factor authentication, have contributed to improvements in authentication security. However, further advancements are required to develop protocols that balance security, computational efficiency, and scalability—particularly in user-server architectures operating over wireless networks or resource-constrained IoT systems, where session key establishment and lightweight operations are crucial for practicality.

Moreover, most existing solutions primarily focus on either improving cryptographic strength or reducing overhead, yet few thoroughly address the need for robust anonymity and resilience against offline password guessing. Thus, there is a pressing need for a new lightweight user-server authentication protocol that offers strong security features (e.g., forward secrecy, replay prevention, and impersonation resistance) while minimizing computational and communication costs. In the next sections, we propose such a protocol and illustrate how it meets these requirements through formal and informal security analyses, as well as performance evaluations.

**Table 1**  
Survey of Recent Research in Secure Authentication

References	Types	Weaknesses	Year
Li et al. (2018)	Lightweight	De-synchronization attack	2018
Chandrakar and Om (2018)	ECC-based	Offline-password guessing attack, Spoofing attack	2018
Zheng et al. (2018)	RFID	Replay attack, Impersonation attack	2018
Xiang and Zheng (2020)	Lightweight	Session key disclosure, Stolen smart card attack, Impersonation attacks	2020
Oh et al. (2021)	Lightweight	-	2021
Gupta et al. (2019)	Lightweight	Privileged-insider attack, Compromised sensing devices, De-synchronization attack	2019
Xu et al. (2019)	Lightweight	Key compromise attack, Replay attack, Impersonation attack	2019
Alzahrani et al. (2021)	Lightweight	-	2021
Wang et al. (2022)	Ultra-lightweight	Secret disclosure attack, De-synchronization attack	2022
Aghili et al. (2019a)	Lightweight	Traceability attack, Secret value disclosure	2019
Gabsi et al. (2021)	ECC-based	Violation of tag anonymity, Traceability attack, Forward and backward	2021
Challa et al. (2018)	ECC-based	Replay attack, DoS attack, Forgery attack	2018
Amin et al. (2018)	ECC-based	Denial of Service attack, Message tampering attack	2018
Sureshkumar et al. (2019)	ECC-based	Tracking attack, De-synchronization vulnerability, Integrity inconsistency	2019
Saeed et al. (2018)	Lightweight	Forged certificateless signature attack	2018
Jia et al. (2022)	Lightweight	Privacy concerns	2022
Yang et al. (2022)	ECC-based	-	2022
Servati and Safkhani (2023)	ECC-based	-	2023
Hosseinzadeh et al. (2024)	Lightweight	-	2024
Huang (2024)	ECC-based	-	2024
Wang et al. (2024)	ECC-based	-	2024
Servati et al. (2025)	ECC-based	-	2025

### 3. System Architecture and Security Considerations

This section provides an overview of the architecture and operational flow of user-server authentication in IoT-driven healthcare environments. In modern healthcare systems, medical data is continuously collected, processed, and exchanged between users and medical servers. Patients, healthcare providers, and institutions rely on secure user-server interactions to ensure authorized access to critical health information while preserving data integrity and confidentiality. As shown in Figure 3, patient records, including vital signs such as heart rate, blood pressure, and body temperature, are frequently transmitted between authenticated users and remote medical servers, facilitating real-time patient monitoring, clinical

diagnostics, and remote healthcare services.

The authentication process involves two primary entities: the *user* (patients, doctors, or healthcare personnel) and the *server* (healthcare providers, cloud storage, or medical service platforms). The user must authenticate with the healthcare server before requesting or modifying any medical information. Secure authentication ensures that only legitimate users with valid credentials can access confidential patient data while preventing unauthorized access attempts. The authentication server verifies user credentials, issues secure session tokens, and facilitates encrypted data exchanges between authenticated parties.

However, direct communication between users and medical servers over public or semi-secure networks introduces significant security and privacy risks. Unauthorized access,



identity spoofing, session hijacking, and data manipulation attacks threaten patient confidentiality and the accuracy of medical records. Ensuring mutual authentication between users and healthcare servers is critical to preventing adversaries from intercepting, modifying, or forging transmitted medical data. Additionally, session key leakage during authentication must be prevented to avoid unauthorized decryption of sensitive healthcare communications.

Many healthcare systems employ cloud-based data storage, providing long-term accessibility for authorized personnel. While cloud platforms enhance scalability and remote access, they also introduce vulnerabilities such as unauthorized data retrieval, session hijacking, and eavesdropping attacks. Traditional authentication mechanisms often fail to adequately mitigate these risks, highlighting the need for a lightweight yet robust authentication protocol that guarantees data confidentiality, integrity, and access control.

To address these security concerns, we propose a new authentication protocol designed to establish a secure framework for user-server authentication in IoT-based healthcare environments. The proposed scheme ensures mutual authentication between users and remote medical servers, verifying the legitimacy of both entities before granting access to sensitive medical records. Additionally, a secure session key establishment mechanism is integrated, ensuring that encrypted communications remain confidential and resistant to interception attacks.

The key security features of the proposed authentication protocol include:

- **Mutual Authentication:** Both the user and the server verify each other's authenticity before initiating communication.
- **Anonymity and Untraceability:** The authentication mechanism prevents adversaries from linking authentication requests to specific users.
- **Session Key Establishment:** A secure cryptographic session key is generated for each authenticated session, ensuring data confidentiality and integrity.
- **Resistance Against Attacks:** The protocol mitigates identity spoofing, replay attacks, session hijacking, man-in-the-middle attacks, and eavesdropping threats.
- **Lightweight Computation:** Optimized cryptographic operations minimize processing overhead, making the protocol suitable for resource-constrained IoT devices in healthcare environments.

By integrating formal security validation techniques and an optimized authentication structure, the proposed protocol ensures that patient data remains protected across various healthcare settings, including hospitals, telemedicine platforms, and remote patient monitoring systems.

## 4. Proposed Protocol: LAPWN

The proposed authentication protocol consists of multiple phases, ensuring secure communication and data integrity

between users and the authentication server. The following sections outline these phases in detail.

### 4.1. Initialization Phase

During this phase, the system administrator (*SA*) initializes and configures each component of the authentication framework, guaranteeing that every registered user is assigned a distinct credential, which is securely maintained on the authentication server. Unlike authentication schemes based on elliptic curve cryptography (ECC), the proposed protocol relies on a cryptographically robust, collision-resistant one-way hash function to ensure secure authentication.

A collision-resistant one-way hash function is a deterministic mapping, represented as  $H : 0, 1 \rightarrow 0, 1^k$ , which transforms an input of arbitrary length  $x \in 0, 1$  into a fixed-size output of  $k$  bits. Its security is based on the assumption that an adversary  $\mathcal{A}$  has an extremely low probability of successfully identifying two distinct messages,  $m$  and  $n$ , that produce the same hash value, i.e.,  $H(m) = H(n)$ , thereby making hash collisions computationally infeasible.

The advantage of an adversary  $\mathcal{A}$  in breaking the hash function is defined as:

$$\text{Adv}_{\mathcal{A}}^{\text{Hash}}(t) = \Pr [(m, n) \in_R \mathcal{A} : m \neq n, H(m) = H(n)] \quad (1)$$

where  $\mathcal{A}$  selects a pair  $(m, n)$  and computes the probability of finding a collision within polynomial execution time  $t$ . The security assumption states that:

$$\text{Adv}_{\mathcal{A}}^{\text{Hash}}(t) \leq \epsilon \quad (2)$$

where  $\epsilon > 0$  is an insignificant probability, ensuring that the likelihood of successfully breaking the hash function within time  $t$  remains negligible Das et al. (2018).

During the initialization phase, the system administrator (*SA*) assigns unique credentials to each user and hashes sensitive identifiers before storing them on the authentication server. This ensures that even if an attacker gains unauthorized access to stored credentials, the irreversibility of the hash function prevents them from reconstructing the original authentication data.

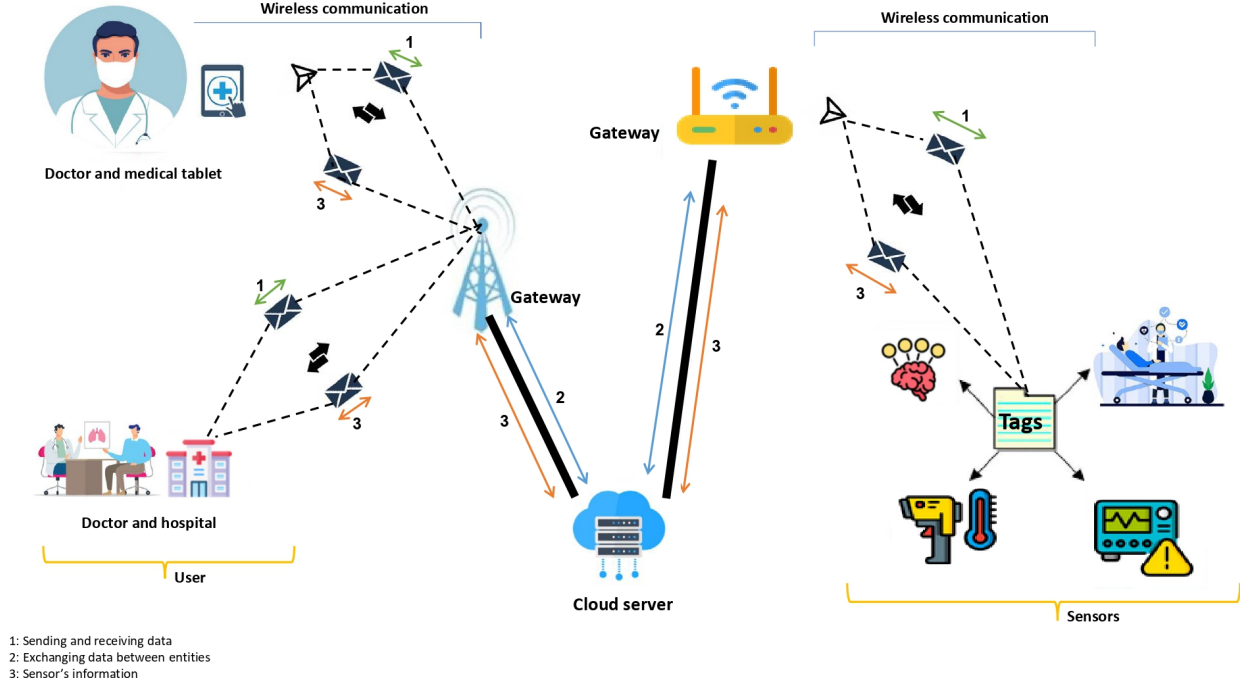
To enhance security, the system employs salting and iterative hashing techniques, ensuring that even identical user credentials result in distinct hash values, thereby mitigating precomputed attacks such as rainbow table attacks.

The notations used throughout this paper are defined in Table 2, outlining key parameters in the authentication scheme.

### 4.2. User Enrollment Phase

The user enrollment phase ensures a secure registration process between the user and the authentication server, maintaining data confidentiality and integrity. The process follows these steps:

1. The user selects an identifier  $ID_i$  and a password  $PW_i$ , then generates a secret key  $K_i \in F_q$ . Using these values, the user computes the hashed representations



**Figure 1:** Medical Wireless Sensor Networks (MWSN): Structural Overview.

as  $HID_i = h(K_i \| ID_i)$  and  $HPW_i = h(K_i \| PW_i)$ . The computed values,  $\{HID_i, HPW_i\}$ , are securely transmitted to the authentication server over a protected communication channel.

2. Upon receiving the enrollment request, the authentication server generates a system-specific secret key  $K_j \in F_q$  and computes  $RID_i = h(HID_i \oplus K_j)$  and  $SB_i = h(HPW_i \oplus K_j)$ . It then derives additional authentication parameters:  $A_i = RID_i \oplus h(HPW_i \| TC_{ui})$ ,  $B_i = SB_i \oplus h(HID_i \| TC_{ui})$ ,  $C_i = h(K_j \| TC_{ui})$ , and  $D_i = C_i \oplus h(HID_i \| TC_{ui})$ . The server computes the user's unique session identifier as  $USN_i = h(HID_i \| K_j)$ . The server securely stores  $USN_i$  in its database for future authentication and transmits the values  $\{A_i, B_i, D_i\}$  back to the user.
3. Upon receiving these parameters, the user reconstructs the required values as  $RID'_i = A_i \oplus h(HPW_i \| TC_{ui})$  and  $SB'_i = B_i \oplus h(HID_i \| TC_{ui})$ . Using these values, the user derives the final authentication token as  $UF_i = h(RID'_i \| SB'_i \| K_i)$ . The user then securely stores authentication-related data, including  $\{A_i, B_i, D_i, UF_i\}$ , ensuring that future authentication attempts remain secure.

The overall process of user enrollment is summarized in Algorithm 1.

#### 4.3. Login Phase

In this phase, the user initiates the login process by entering their identifier  $ID_i$ , password  $PW_i$ , and secret key  $K_i$  into

#### Algorithm 1 User Enrollment Phase

- 1: **Step 1: User Registration**
- 2: User selects an identifier  $ID_i$  and password  $PW_i$
- 3: User generates a secret key  $K_i \in F_q$
- 4: Computes  $HID_i = h(K_i \| ID_i)$
- 5: Computes  $HPW_i = h(K_i \| PW_i)$
- 6: Securely transmits  $\langle HID_i, HPW_i \rangle$  to the authentication server
- 7: **Step 2: Server Processing**
- 8: Authentication server selects a secret key  $K_j \in F_q$
- 9: Computes  $RID_i = h(HID_i \oplus K_j)$
- 10: Computes  $SB_i = h(HPW_i \oplus K_j)$
- 11: Derives authentication parameters:
- 12:  $A_i = RID_i \oplus h(HPW_i \| TC_{ui})$
- 13:  $B_i = SB_i \oplus h(HID_i \| TC_{ui})$
- 14:  $C_i = h(K_j \| TC_{ui})$
- 15:  $D_i = C_i \oplus h(HID_i \| TC_{ui})$
- 16: Computes unique session identifier  $USN_i = h(HID_i \| K_j)$  and stores it
- 17: Securely transmits  $\langle A_i, B_i, D_i \rangle$  to the user
- 18: **Step 3: User Stores Enrollment Data**
- 19: Upon receiving data from the server, the user verifies parameters:
- 20: Computes  $RID'_i = A_i \oplus h(HPW_i \| TC_{ui})$
- 21: Computes  $SB'_i = B_i \oplus h(HID_i \| TC_{ui})$
- 22: Computes  $UF_i = h(RID'_i \| SB'_i \| K_i)$
- 23: Securely stores  $\{A_i, B_i, D_i, UF_i\}$  in their device

**Table 2**  
Overview of Notations and Parameters

Symbol	Description
$U_i$	User node
$S_j$	Server node
$ID_i$	User's unique identifier
$PW_i$	User-selected password
$K_i$	User-generated secret key
$K_j$	Server-generated secret key
$Sk$	Secret key computed between entities
$H(.)$	Cryptographic hash function (256 or 160 bits)
$TC_{ui}$	Timestamp used during user enrollment
$TS_x$	Timestamp used for message authentication
$\Delta T$	Time threshold for replay attack prevention
$\oplus$	Bitwise XOR operation
$\parallel$	Concatenation operation
$A$	System administrator
$r_u$	Random nonce generated by user
$r_s$	Random nonce generated by server
$USN_i$	Unique session identifier assigned to the user
$S_{Key}$	Session key established between user and server

the system. The login process consists of the following steps:

1. **Hashing User Credentials:** The user first computes the hashed representations of their identity and password as  $HID_i = h(K_i \parallel ID_i)$  and  $HPW_i = h(K_i \parallel PW_i)$ .
2. **Deriving Session Parameters:** The user then derives the session parameters by computing  $RID_i^* = h(HID_i \oplus K_j) \oplus A_i$  and  $SB_i^* = h(HPW_i \oplus K_j) \oplus B_i$ .
3. **Generating Authentication Verification Token:** The user generates the authentication verification token as  $UF_i^* = h(RID_i^* \parallel SB_i^* \parallel K_i)$ .
4. **Verification:** To verify the correctness of the authentication process, the user compares the computed value  $UF_i^*$  with the previously stored authentication value  $UF_i$ . If they match, the authentication phase successfully begins.

#### 4.4. Authentication Phase

After successful login verification, the user proceeds to the authentication phase. The authentication process consists of the following steps:

1. **User Initiates Authentication:** The user generates a random secret  $r_u$  and includes a timestamp  $TS_1$  to prevent replay attacks. The user then computes  $C_i^* = D_i \oplus h(HID_i \parallel TC_{ui})$  and  $M_1 = r_u \oplus h(C_i^* \parallel TS_1)$ . Additionally, the user derives  $M_2 = ID_i \oplus h(r_u \parallel TS_1)$  and  $M_3 = h(ID_i \parallel r_u \parallel TS_1)$ . The user then transmits  $\langle M_1, M_2, M_3, TS_1 \rangle$  to the server.
2. **Server Verifies the Request:** Upon receiving the authentication request, the server verifies the time synchronization by checking  $|TS_1 - TS_1^*| \leq \Delta T$ . If valid,

the server computes  $C_i' = h(K_j \parallel TC_{ui})$  and extracts  $r_u^* = M_1 \oplus h(C_i' \parallel TS_1)$ . It then reconstructs the user identifier as  $ID_i^* = M_2 \oplus h(r_u^* \parallel TS_1)$  and computes the session identifier  $USN_i^* = h(ID_i^* \parallel K_j)$ .

3. **Server Confirms Authentication:** To confirm the request, the server verifies that  $M_3^* = h(ID_i^* \parallel r_u^* \parallel TS_1)$ . If this holds, authentication is approved, and the server proceeds to generate a random secret  $r_s \in F_q$ . The server also records timestamp  $TS_2$  and computes  $M_4 = r_s \oplus h(ID_i^* \parallel C_i' \parallel TS_2)$  and the session key  $S_{Key_s} = h(r_s \parallel r_u^*)$ . Finally, the server generates a confirmation message  $M_5 = h(ID_i^* \parallel S_{Key_s} \parallel TS_2)$  and transmits  $\langle M_4, M_5, TS_2 \rangle$  to the user.
4. **User Verifies the Server's Response:** Upon receiving the server's response, the user checks the time synchronization by verifying  $|TS_2 - TS_2^*| \leq \Delta T$ . The user then extracts  $r_s^* = M_4 \oplus h(ID_i \parallel C_i^* \parallel TS_2)$  and derives the session key as  $S_{Key_u} = h(r_s^* \parallel r_u)$ . The user then confirms authentication by ensuring  $M_5^* = h(ID_i \parallel S_{Key_u} \parallel TS_2)$ . If  $M_5^* = M_5$ , authentication is successfully completed, ensuring that  $S_{Key_u} = S_{Key_s}$ .

This process establishes mutual authentication and session key agreement, securing further communication between the user and the server. The complete authentication flow is illustrated in Figure 2.

#### 4.5. Update and Change Password Phase

In this phase, the user initiates a password change request by providing their credentials, including the identifier  $ID_i$ ,

password  $PW_i$ , and secret key  $K_i$ . The password update process follows these steps:

1. **User Verification:** The system first verifies the user's identity by computing  $HID_i = h(K_i \| ID_i)$  and  $HPW_i = h(K_i \| PW_i)$ . Then, the system derives session parameters as  $RID_i^* = h(HID_i \oplus K_j)$  and  $SB_i^* = h(HPW_i \oplus K_j)$ . The authentication verification token is generated as  $UF_i^* = h(RID_i^* \| SB_i^* \| K_i)$ . The computed value  $UF_i^*$  is compared with the stored authentication token  $UF_i$ . If they match, the user is successfully authenticated and proceeds to update their credentials.
2. **User Chooses New Credentials:** After successful authentication, the user selects a new identifier  $ID_i^{new}$  or a new password  $PW_i^{new}$ . The system updates the stored credentials by computing  $HID_i^{new} = h(K_i \| ID_i^{new})$  and  $HPW_i^{new} = h(K_i \| PW_i^{new})$ .
3. **Updating the Authentication Parameters:** The system derives the updated authentication parameters as  $RID_i^{new} = h(HID_i^{new} \oplus K_j) \oplus A_i$  and  $SB_i^{new} = h(HPW_i^{new} \oplus K_j) \oplus B_i$ .
4. **Finalizing the Update:** The user computes the new authentication token as  $UF_i^{new} = h(RID_i^{new} \| SB_i^{new} \| K_i)$ . The updated values ensure the security and integrity of the authentication process, completing the password change phase.

The complete procedure for updating the password in the proposed protocol is outlined in Algorithm 2.

## 5. Security Evaluation

### 5.1. Formal and Informal Security Assessment

To comprehensively evaluate the security of the proposed protocol, both formal and informal verification methods are employed. These approaches are widely recognized in cryptographic protocol analysis, ensuring the robustness and resilience of authentication mechanisms against potential threats. For manual verification, well-established analytical models such as the Real-or-Random (RoR) model, GNY logic, and BAN logic are commonly used. These models provide a theoretical foundation for validating essential security properties, including mutual authentication, session key secrecy, and resistance to common cryptographic attacks.

In addition to manual analysis, automated verification tools offer a rigorous security assessment by simulating potential attack scenarios. In this study, we employ Scyther and ProVerif, two well-known formal verification tools, to systematically analyze the security properties of the proposed authentication scheme. The Scyther tool is utilized in two configurations: the standard model and the compromise model, allowing for a comprehensive assessment of protocol security under different adversarial conditions.

Furthermore, the RoR model is adopted as a widely accepted manual approach for formal security validation, providing an additional layer of theoretical proof for the protocol's security guarantees. By integrating these analytical techniques,

---

### Algorithm 2 Password Update Process in the Proposed Protocol

---

**Data:** User credentials  $\langle ID_i, PW_i, K_i \rangle$

**Result:** Successfully updated password

---

- 1: **Step 1: User Authentication**
  - 2:  $U_i$  enters their identifier  $ID_i$ , password  $PW_i$ , and secret key  $K_i$ .
  - 3: The system computes:
  - 4:  $HID_i = h(K_i \| ID_i)$ ,  $HPW_i = h(K_i \| PW_i)$
  - 5:  $RID_i^* = h(HID_i \oplus K_j)$ ,  $SB_i^* = h(HPW_i \oplus K_j)$
  - 6:  $UF_i^* = h(RID_i^* \| SB_i^* \| K_i)$
  - 7: **if**  $UF_i^* == UF_i$  **then**
  - 8: **Step 2: User Chooses New Credentials**
  - 9:  $U_i$  selects a new identifier  $ID_i^{new}$  or password  $PW_i^{new}$ .
  - 10: The system computes:
  - 11:  $HID_i^{new} = h(K_i \| ID_i^{new})$
  - 12:  $HPW_i^{new} = h(K_i \| PW_i^{new})$
  - 13:  $RID_i^{new} = h(HID_i^{new} \oplus K_j) \oplus A_i$
  - 14:  $SB_i^{new} = h(HPW_i^{new} \oplus K_j) \oplus B_i$
  - 15:  $UF_i^{new} = h(RID_i^{new} \| SB_i^{new} \| K_i)$
  - 16: The system updates stored credentials with the new values.
  - 17: **Return:** Password update successful.
  - 18: **else**
  - 19: **Step 3: Authentication Failure**
  - 20: The session is terminated.
  - 21: **Return:** Password update failed.
  - 22: **end if**
- 

we demonstrate that the proposed protocol effectively mitigates various attacks, including impersonation, replay, man-in-the-middle (MITM), and offline password guessing attacks, while maintaining efficiency in computational and communication overhead.

The combined use of formal verification tools and theoretical models ensures a robust and well-founded security evaluation, reinforcing the reliability of the proposed authentication mechanism in real-world applications.

### 5.2. Informal Security Analysis

The proposed authentication protocol is designed with a robust security framework to withstand various types of attacks that commonly target authentication schemes. This section presents an informal security analysis by evaluating potential attack vectors and demonstrating the protocol's resilience.

#### 5.2.1. User Impersonation Attack

In the proposed scheme, messages  $M_1$ ,  $M_2$ , and  $M_3$  are transmitted over a public channel, which an adversary  $\mathcal{A}$  could potentially intercept. However,  $\mathcal{A}$  cannot fabricate valid authentication messages without knowledge of the secret random value  $r_u$ , which is securely embedded in the cryptographic operations. Since  $r_u$  remains undisclosed, an adversary cannot generate valid authentication requests, effectively preventing user impersonation attacks.





#### 5.2.4. Replay Attack

The proposed protocol prevents replay attacks by incorporating session-specific nonces and timestamps into messages  $M_1$  through  $M_5$ . These time-dependent elements ensure that each authentication session is unique. Any attempt to reuse previously intercepted messages is detected and rejected by the protocol, preventing adversaries from exploiting replayed authentication requests.

#### 5.2.5. Man-in-the-Middle (MITM) Attack

In a MITM attack, an adversary attempts to intercept and manipulate communication between two parties. However, in the proposed protocol, authentication messages are cryptographically linked to secret values that the adversary does not possess. For example, when  $U_i$  sends  $\{M_1, M_2, M_3, TS_1\}$ , the server verifies  $M_3^*$  using  $K_j$ , which remains unknown to  $\mathcal{A}$ . Since  $\mathcal{A}$  cannot alter these values without detection, the protocol ensures message integrity and prevents MITM attacks.

#### 5.2.6. Session-Specific Temporary Information Attack

To protect against session-specific information attacks, the protocol employs fresh nonces  $r_u$  and  $r_s$  in each session. The session key is derived as  $s_{key} = h(r_s \| r_u)$ , ensuring that even if an adversary obtains temporary secrets from one session, they cannot compute the session key for subsequent communications. This guarantees that past and future sessions remain cryptographically isolated.

#### 5.2.7. Offline Identity/Password Guessing Attack

The protocol is resilient against offline guessing attacks, even in scenarios where an adversary gains access to stored parameters  $\{A_i, B_i, D_i, UF_i^*\}$ . To perform a successful guessing attack,  $\mathcal{A}$  must correctly guess both  $ID_i$  and  $PW_i$ , which are cryptographically linked to  $K_i$ . Since the probability of guessing these values simultaneously is negligible, the protocol effectively resists offline credential guessing attempts.

#### 5.2.8. Online Identity/Password Guessing Attack

The proposed scheme protects against online guessing attacks by ensuring that publicly transmitted messages  $\{M_1, M_2, M_3\}$  do not directly expose password-dependent values. For instance, in  $M_2 = ID_i \oplus h(r_u \| TS_1)$ , the identifier  $ID_i$  is concealed using a nonce. This cryptographic masking prevents an adversary from isolating and brute-forcing the user's identity or password, safeguarding against online guessing threats.

#### 5.2.9. Mutual Authentication

The protocol achieves mutual authentication by verifying both user and server identities. When  $U_i$  sends the login message  $M_3$ , the server authenticates the user by checking the validity of  $M_3^*$ . Conversely, the server authenticates itself to the user through the verification of  $M_5$  and  $M_5^*$ , ensuring that both parties confirm each other's legitimacy before further communication.

#### 5.2.10. Forward Secrecy

The proposed protocol ensures forward secrecy, meaning that even if the server's secret key  $K_j$  is compromised, previous session keys remain secure. Since the session key is derived as  $s_{key} = h(r_s^* \| r_u)$  using ephemeral values that change in every session, past communications are protected from future key disclosures.

#### 5.2.11. User Anonymity

User anonymity is preserved by ensuring that the user's identifier  $ID_i$  is not transmitted in plaintext during authentication. Instead,  $ID_i$  is cryptographically embedded within secure exchanges, preventing adversaries from tracking users based on intercepted authentication messages  $\{M_1, M_2, M_3\}$ . This ensures user privacy and prevents identity-based tracking.

Thus, from this informal analysis, we see that the proposed protocol mitigates major threats such as impersonation, replay, MITM, DoS, and password guessing, thereby providing a robust security architecture for user-server interactions.

### 5.3. Formal security assessment

#### 5.4. Formal Security Analysis Using Scyther

Conducting a formal security analysis is essential for validating the robustness of cryptographic protocols. One of the most widely utilized tools for such evaluations is Scyther, which enables rigorous formal verification based on the well-established Dolev-Yao threat model Dolev and Yao (1983). This tool systematically explores protocol executions, identifying potential security violations by modeling interactions between protocol participants within an adversarial environment.

Scyther employs the Security Protocol Description Language (SPDL) to define roles, exchanged messages, and security assertions. By simulating all possible execution paths, it evaluates compliance with predefined security objectives. If vulnerabilities exist, Scyther generates attack traces, visually demonstrating how an adversary could exploit protocol weaknesses. The tool verifies critical security properties, including:

- **Secrecy:** Ensuring that sensitive data remains protected against adversaries.
- **Authentication:** Verifying that only legitimate entities participate in the communication.
- **Message Integrity:** Preventing unauthorized alterations to transmitted messages.

The Scyther framework operates under standard cryptographic assumptions, treating cryptographic primitives as black-box functions while ensuring message integrity. It models protocols as structured role-based interactions, where entities exchange messages according to a predefined logic. This symbolic approach enables comprehensive security verification against various attack scenarios.

In this study, Scyther was employed to evaluate the proposed authentication protocol, ensuring its resilience against adversarial threats. The protocol was modeled with two primary entities—the user and the server—representing key interactions in the authentication process. The security claims verified in this analysis include data confidentiality, mutual authentication, and resistance to replay and impersonation attacks.

To further strengthen the evaluation, two variants of Scyther were utilized:

- **Standard Scyther:** Analyzes the protocol under conventional adversarial assumptions.
- **Compromise Scyther:** Introduces extended adversarial capabilities, allowing attackers to compromise long-term keys, perform key-reveal attacks, and manipulate partial protocol executions.

The results confirmed that the proposed protocol remains secure even under enhanced adversarial conditions. Figures 3 and 4 depict the configuration settings for both the compromise and standard modes in Scyther, emphasizing the enhanced attack capabilities enabled in the compromise model. The security assessment of the proposed protocol under compromise mode, particularly evaluating its resilience against Post-Compromise Forward Secrecy (PFC) and Session Key Reveal attacks, is illustrated in Figures 5 and 6, respectively. In contrast, the security validation results obtained using the standard mode of Scyther are presented in Figure 7.

The findings from this formal evaluation confirm that the proposed protocol maintains strong security guarantees, effectively mitigating threats such as replay attacks, key compromise impersonation, and session key disclosure while ensuring secure mutual authentication between the communicating entities.

#### 5.4.1. Security Verification Using ProVerif

To rigorously assess the security of the proposed authentication protocol, we employed ProVerif, a widely used automated verification tool for cryptographic protocols. ProVerif is particularly suitable for security analysis due to its ability to symbolically model cryptographic mechanisms, including hash functions, encryption schemes, and key exchange protocols. The tool systematically examines critical security properties such as confidentiality, integrity, and mutual authentication under adversarial conditions.

In the verification process, the enrollment phase was modeled over secure channels to ensure the confidentiality of long-term credentials, while the authentication and key agreement phases were simulated over public channels, reflecting real-world attack scenarios. ProVerif's adversarial model, based on the Dolev-Yao threat model, assumes an active attacker with full control over public communication links, enabling a rigorous security assessment.

The ProVerif analysis primarily focused on validating core security properties, including mutual authentication, session key secrecy, and resistance to impersonation attacks. The

verification results, as shown in Figure 8, confirm that the proposed protocol meets these essential security requirements. The verification summary highlights that the authentication scheme effectively upholds fundamental security objectives:

1. **Confidentiality of Weak Secrets:** The protocol ensures that critical identifiers and session-related secrets remain protected. The verification output in Figure 8 confirms that sensitive parameters such as  $ID_i$ ,  $PW_i$ ,  $k_i$ ,  $r_u$ , and  $r_s$  remain uncompromised. This guarantees that adversaries cannot extract or manipulate these values, preserving data secrecy.
2. **Prevention of Unauthorized Access:** The results confirm that the authentication mechanism effectively mitigates unauthorized access attempts. Queries such as `not attacker(uru[!1 = v])` and `not attacker(srs[T2_1 = v, A6_1 = v_1, A2_2 = v_2, !1 = v_3])` indicate that the protocol prevents adversaries from gaining control over sensitive authentication values. This strengthens its resilience against replay and key-compromise attacks.
3. **Injection and Event-Based Security Guarantees:** The security verification also validates the integrity of user-server interactions. The queries `inj-event(endUserA) ==> inj-event(beginUserA)` and `inj-event(endServer) ==> inj-event(beginServer)` confirm that authentication is correctly enforced, preventing adversarial interference during session initiation and completion. This ensures that each authentication session executes securely without interception or manipulation.

Based on these findings, the proposed authentication protocol ensures robust security guarantees, preventing identity theft, replay attacks, and unauthorized key disclosure. The ProVerif verification results provide strong evidence that the protocol upholds essential cryptographic security principles, offering secure entity authentication, confidentiality, and resistance to active adversarial threats.

#### 5.4.2. Formal Security Analysis Using the RoR Model

The Real-Or-Random (RoR) model serves as a fundamental security framework for assessing whether an adversary  $\mathcal{A}$  can effectively distinguish between an actual session key and a randomly generated one. This evaluation measures the protocol's resilience against session key disclosure and key-compromise impersonation attacks.

The core concept behind RoR security is that, within a well-designed protocol, even an adversary with substantial computational power should have no advantage in differentiating a real session key from a random one beyond a probability close to  $\frac{1}{2}$ . If an adversary can reliably exceed this threshold, the protocol is deemed insecure.

#### 5.4.3. RoR Model

The proposed protocol consists of two main entities: the user ( $U_i$ ), and the authentication server ( $S_j$ ). We define  $\Pi_{U_i}^a$  and  $\Pi_{S_j}^b$  to represent the instances of  $U_i$  and  $S_j$  participating in

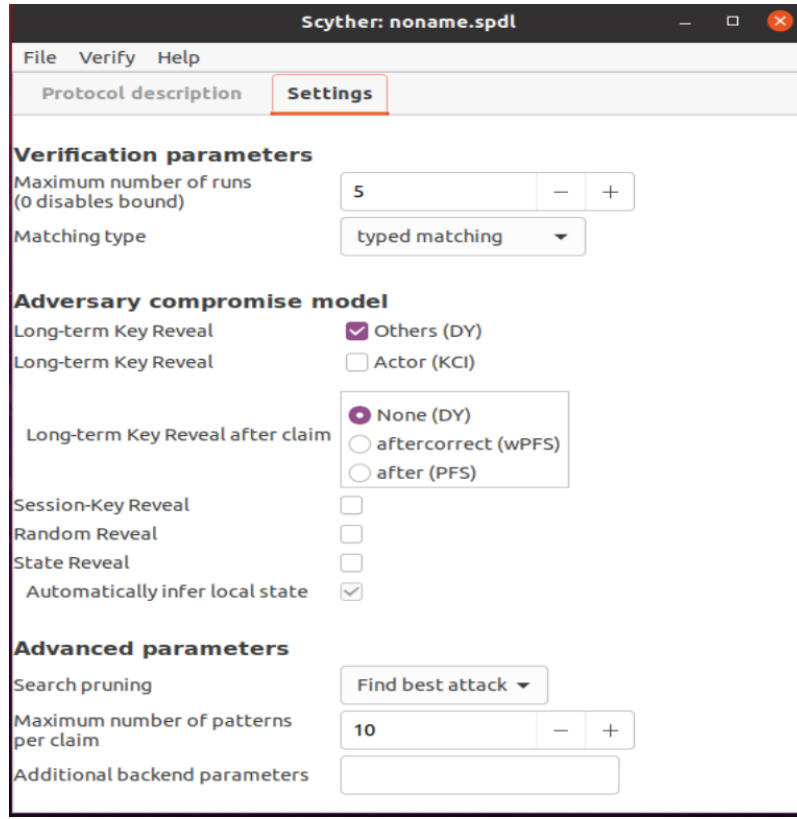


Figure 3: Configuration Panel of Scyther (Compromise-0.9.2 Version)

the protocol execution. An adversary  $\mathcal{A}$  is assumed to have access to the following cryptographic queries:

- **Execute**( $\Pi_{U_i}^a, \Pi_{S_j}^b$ ): The adversary  $\mathcal{A}$  passively eavesdrops on the communication between  $U_i$  and  $S_j$ , capturing all exchanged messages.
- **Send**( $\Pi_{U_i}^a, M$ ): The adversary  $\mathcal{A}$  actively injects message  $M$  to a session instance of  $U_i$  and observes the corresponding response.
- **Hash**(String): The adversary  $\mathcal{A}$  computes the hash of any given string using the protocol's cryptographic hash function.
- **Corrupt**( $U_i$  or  $S_j$ ): The adversary  $\mathcal{A}$  compromises either a user or a server and obtains all stored cryptographic material, including long-term secrets and session-specific values.
- **Test**( $\Pi_{U_i}^a$ ): A challenge query where  $\mathcal{A}$  attempts to distinguish between a real session key and a randomly generated value.

The adversary's goal is to determine whether it received a real session key or a random string. The probability of a successful attack is denoted as  $Adv_{\mathcal{A}}^{\text{Protocol}}$ .

#### 5.4.4. Security Theorem

**Theorem 1:** Under the assumption that the hash function  $H(\cdot)$  is collision-resistant and that the random values  $r_u$  and  $r_s$  are independently chosen for each session, the probability that an adversary  $\mathcal{A}$  can break the session key secrecy of the proposed protocol is bounded by:

$$Adv_{\mathcal{A}}^{\text{Protocol}} \leq \frac{q_h^2}{|H|} + \frac{q_s}{2^{t-1}|D|} \quad (3)$$

where:

- $q_h$  denotes the number of hash function queries made by  $\mathcal{A}$ .
- $q_s$  represents the total number of session instances executed.
- $|H|$  refers to the size of the hash function's output space.
- $|D|$  denotes the dictionary size for password guessing attacks.
- $t$  is the bit-length of user-specific authentication credentials (e.g., password-derived values).

This bound indicates that the adversary's advantage remains negligible if  $q_h$  and  $q_s$  are constrained within practical limits and if sufficiently strong cryptographic hash functions are used.



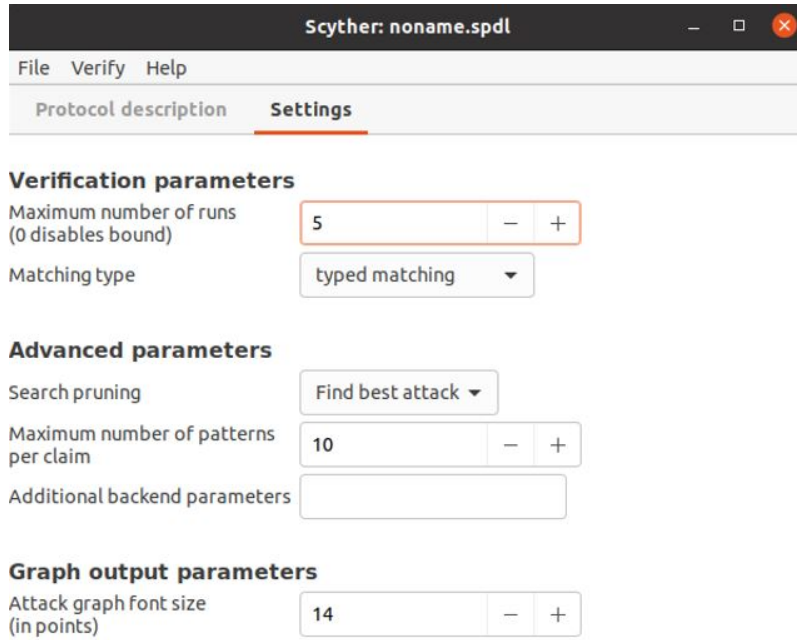


Figure 4: Configuration Panel of Scyther (Standard Version)

Scyther results : verify					
Claim				Status	Comments
userandserver	User	userandserver,User	Secret ru	Ok	No attacks within bound
		userandserver,User1	Nisynch	Ok	No attacks within bound
		userandserver,User2	Alive	Ok	No attacks within bound
		userandserver,User3	Weakagree	Ok	No attacks within bound
	Server	userandserver,Server	Secret rs	Ok	No attacks within bound
		userandserver,Server1	Nisynch	Ok	No attacks within bound
		userandserver,Server2	Alive	Ok	No attacks within bound
		userandserver,Server3	Weakagree	Ok	No attacks within bound
		Done.			

Figure 5: Comprehensive Security Assessment of the LAPWN Authentication Protocol Using Scyther Compromise Mode with the "After PFC" Configuration)

#### 5.4.5. Game-Based Security Proof

The security proof follows a sequence of indistinguishable games  $GM_i (i = 0, 1, 2, 3)$ , where the probability of an adversary's success is denoted as  $Pr[Succ_A^{GM_i}]$ .

- **$GM_0$  (Initial Guessing Game):** The adversary  $\mathcal{A}$  attempts to guess the session key  $Sk$  without interacting with the protocol. The probability of success in this scenario is purely random:

$$Adv_{\mathcal{A}}^{\text{Protocol}} = |2Pr[Succ_A^{GM_0}] - 1| \quad (4)$$

- **$GM_1$  (Passive Attack):** The adversary eavesdrops on

authentication messages  $M_1, M_2, M_3, M_4, M_5$ . However, since  $Sk = h(r_s || r_u)$  depends on fresh random values,  $\mathcal{A}$  cannot reconstruct it. Therefore:

$$Pr[Succ_A^{GM_1}] = Pr[Succ_A^{GM_0}] \quad (5)$$

- **$GM_2$  (Active Attack):** The adversary actively injects messages into the protocol, attempting to modify authentication exchanges. However, since cryptographic hash functions ensure the integrity of exchanged values, the probability of success is upper-bounded by:

Scyther results : verify

Claim				Status	Comments
userandserver	User	userandserver,User	Secret ru	ok	No attacks within bound
		userandserver,User1	Nisynch	ok	No attacks within bound
		userandserver,User2	Alive	ok	No attacks within bound
		userandserver,User3	Weakagree	ok	No attacks within bound
	Server	userandserver,Server	Secret rs	ok	No attacks within bound
		userandserver,Server1	Nisynch	ok	No attacks within bound
		userandserver,Server2	Alive	ok	No attacks within bound
		userandserver,Server3	Weakagree	ok	No attacks within bound

Done.

**Figure 6:** Thorough Security Validation of the Proposed Authentication Scheme Using Scyther Compromise Mode Under the “Session Key Reveal” Configuration

Scyther results : verify

Claim				Status	Comments
userandserver	User	userandserver,User	Secret ru	Ok	No attacks within bound
		userandserver,User1	Nisynch	Ok	No attacks within bound
		userandserver,User2	Alive	Ok	No attacks within bound
		userandserver,User3	Weakagree	Ok	No attacks within bound
	Server	userandserver,Server	Secret rs	Ok	No attacks within bound
		userandserver,Server1	Nisynch	Ok	No attacks within bound
		userandserver,Server2	Alive	Ok	No attacks within bound
		userandserver,Server3	Weakagree	Ok	No attacks within bound

Done.

**Figure 7:** Security assessment of the proposed authentication protocol through the standard version of the Scyther tool

$$Pr[Succ_{\mathcal{A}}^{GM_2}] - Pr[Succ_{\mathcal{A}}^{GM_1}] \leq \frac{q_h^2}{2|H|} \quad (6)$$

- **$GM_3$  (Compromise Attack):** If  $\mathcal{A}$  compromises a user’s device, it gains access to stored parameters  $\{A_i, B_i, D_i, U F_i\}$ . However, breaking the session key secrecy requires inverting the hash function, which is infeasible. The probability of success is thus:

$$Pr[Succ_{\mathcal{A}}^{GM_3}] - Pr[Succ_{\mathcal{A}}^{GM_2}] \leq \frac{q_s}{2^{t-1}|D|} \quad (7)$$

Finally, the overall adversary advantage is given by:

$$Adv_{\mathcal{A}}^{\text{Protocol}} \leq \frac{q_s}{2^{t-1}|D|} + \frac{q_h^2}{|H|} \quad (8)$$

Since this probability remains negligible under practical security parameters, we conclude that the proposed protocol achieves session key secrecy and is resistant to adversarial attacks under the RoR model.

## 6. Implementation of the Proposed Authentication Protocol

This section presents a detailed implementation of the proposed authentication protocol, designed to establish secure authentication between a user and a database server. The protocol has been implemented in Python 3.9, utilizing cryptographic techniques to enhance security and efficiency. The system operates within a graphical user interface (GUI), facilitating seamless interaction for user registration, authentication, and secure communication.

Figure 9 illustrates the development environment within the Thonny IDE, where the implementation has been deployed.

### Verification summary:

- Weak secret IDi is **true**.
- Weak secret PWi is **true**.
- Weak secret ki is **true**.
- Weak secret ru is **true**.
- Weak secret rs is **true**.
- Query inj-event(endUserA)  $\implies$  inj-event(beginUserA) is **true**.
- Query inj-event(endServer)  $\implies$  inj-event(beginServer) is **true**.

**Figure 8:** Security validation of the proposed authentication scheme via ProVerif analysis

The GUI allows users to register and authenticate securely, ensuring a user-friendly experience.

Figure ?? illustrates the user registration process, where users enter their login credentials. The system processes this information through cryptographic hashing and key derivation functions to generate secure authentication parameters before storing them in the database. This ensures resistance against offline attacks and unauthorized access. Figure 11 showcases the login process, requiring users to provide valid authentication credentials. The system retrieves the stored authentication parameters and verifies them against the input data. If authentication is successful, the system proceeds to establish a secure session between the user and the database server.

Figure 12 illustrates the authentication mechanism, where a challenge-response protocol verifies the user's legitimacy and ensures mutual authentication with the database server. A session key, derived through cryptographic operations, secures subsequent communications, while timestamps mitigate replay attacks by maintaining the freshness and validity of authentication messages. Additionally, Figure 12 depicts the secure data exchange process between the authenticated user and the server, where the established session key guarantees data confidentiality and integrity, protecting against eavesdropping and tampering.

Table 3 provides an overview of the authentication process, detailing cryptographic transformations at each stage. It also highlights securely stored parameters, such as hashed user credentials and authentication keys, which enhance the robustness of the authentication mechanism.

To further enhance security, stronger password hashing techniques such as PBKDF2, bcrypt, or Argon2 can be incorporated, significantly increasing resistance to brute-force attacks and ensuring the protection of stored credentials against potential threats.

## 7. Assessment and comparison

### 7.1. Security and Performance Evaluation: Comparative Analysis of the Proposed Authentication Protocol

This section presents a comprehensive comparison between the proposed authentication protocol and other recently de-

veloped schemes. The evaluation considers multiple critical aspects, including security robustness, communication overhead, computational efficiency, and storage requirements. By assessing these factors, we aim to highlight the protocol's strengths and its suitability for resource-constrained environments.

In terms of communication efficiency, the study evaluates the number of exchanged messages required to complete authentication and session establishment. A lower communication overhead is particularly beneficial for IoT environments where bandwidth is limited.

Computational efficiency is analyzed based on the cryptographic operations involved in authentication, including hashing, XOR operations, and modular arithmetic. Protocols that minimize computational complexity are more suitable for devices with constrained processing power.

The results of this comparative analysis provide valuable insights into the trade-offs between security and performance, demonstrating the effectiveness of the proposed protocol in achieving a balance between robust protection and computational feasibility.

### 7.2. Evaluating Security Robustness: The Proposed Authentication Protocol vs. Contemporary Authentication Methods

We perform a comprehensive security assessment of the proposed authentication protocol in comparison to several previously developed schemes, including those presented in Aghili et al. (2019b); Li et al. (2019); Fotouhi et al. (2020); Amintoosi et al. (2022); Fariss et al. (2022); Wu et al. (2023); Ali and Ahmed (2024). Our findings demonstrate that the proposed scheme successfully counteracts numerous security threats, such as replay attacks, privilege escalation, session key leakage, traceability risks, forward secrecy violations, and de-synchronization threats. Conversely, the aforementioned authentication mechanisms remain susceptible to these vulnerabilities. A detailed security evaluation is outlined in Table 4, which highlights the superior resilience of the proposed protocol when juxtaposed with other contemporary authentication models.

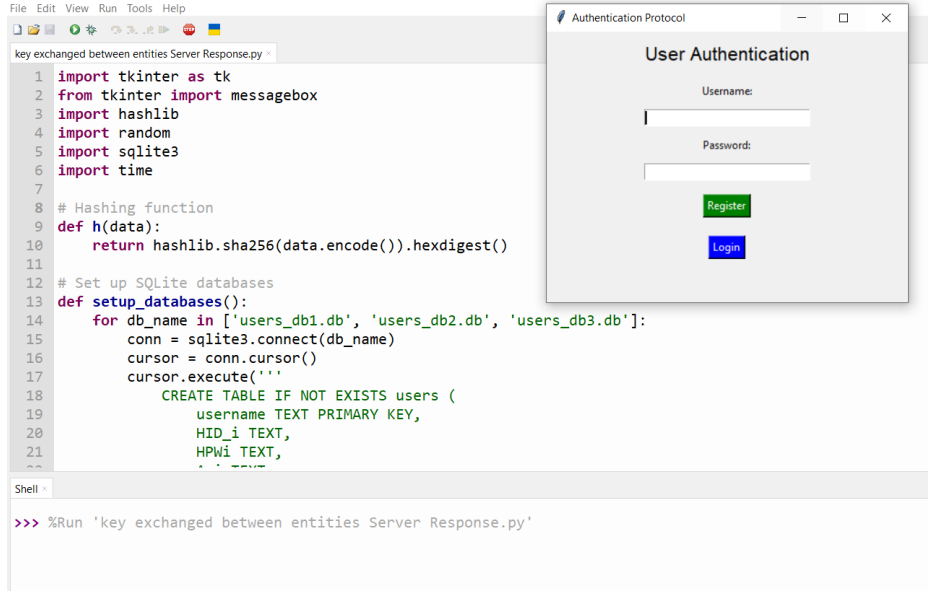


Figure 9: User-Server Secure Authentication and Session Establishment in Thonny IDE

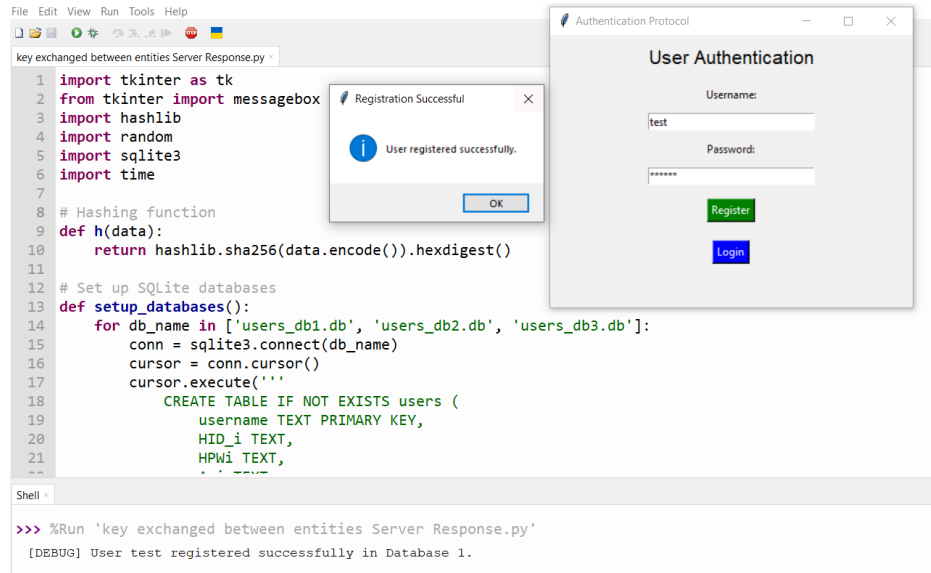


Figure 10: Visual Overview of User Registration in the Developed Authentication System

### 7.3. Comparison of Communication and Computational Overhead

**Performance Evaluation.** Based on the findings in Servati et al. (2025), the approximate execution times for the hash function ( $T_{hf}$ ) and encryption/decryption operations ( $T_{en/d}$ ) are 3 ms and 3.7 ms, respectively. As detailed in Table 5, our proposed authentication protocol incurs a total computational overhead of  $20T_{hf}$  (i.e., 60 ms when  $T_{hf} = 3$  ms). A comparative assessment of this computational efficiency relative to other authentication mechanisms is provided in Table 6, with the results visualized in Figure 13.

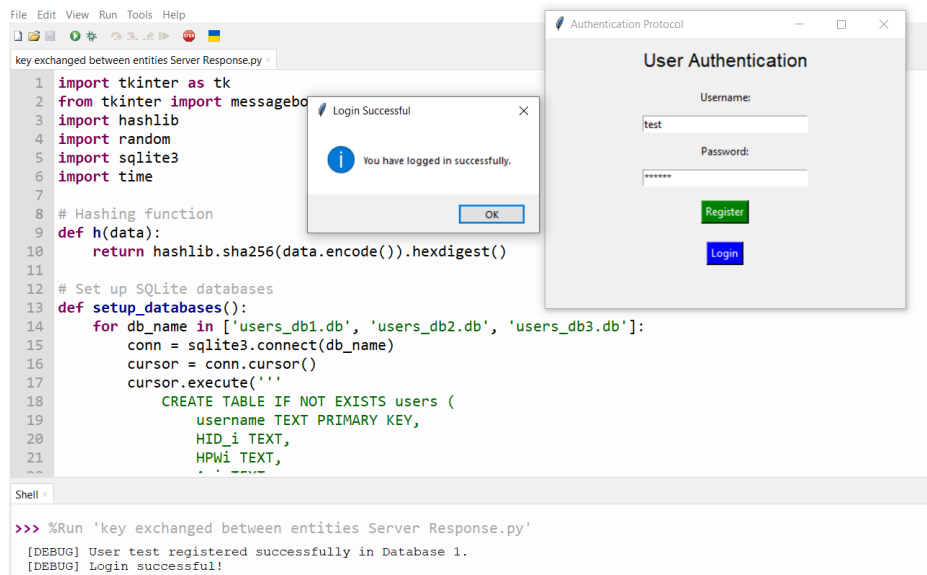
Additionally, we evaluate the communication overhead by measuring the total number of bits exchanged during the authentication process. As outlined in Table 5, the protocol's

communication cost comprises 128 bits for identity transmission, 32 bits for timestamp validation, 128 bits for encryption/decryption, 256 bits for random values, and 256 bits for hash function outputs. A comparative analysis of our scheme's communication overhead, presented in Table 7 and visualized in Figure 14, demonstrates that the proposed authentication scheme achieves a balanced trade-off between security guarantees and communication efficiency compared to existing protocols.

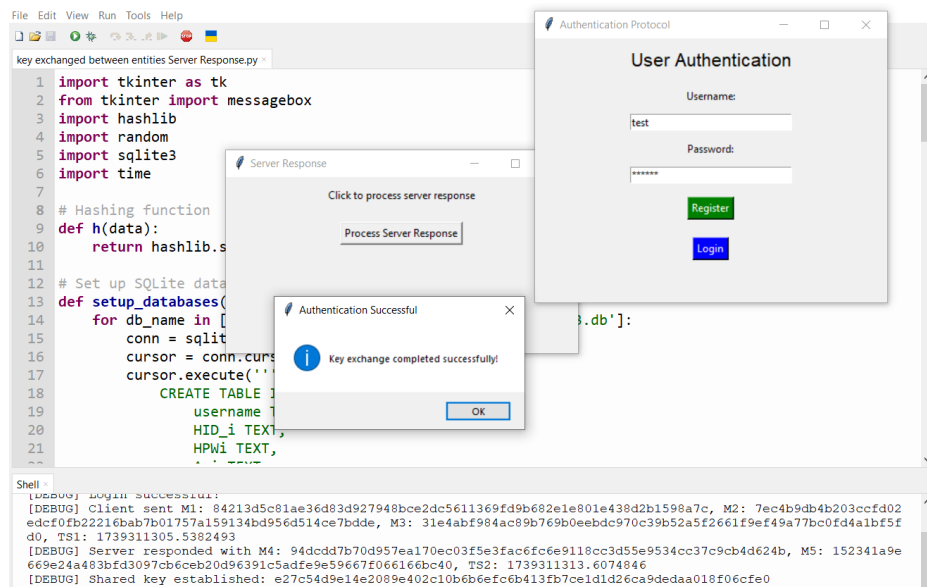
## 8. Conclusion

This research introduces a robust authentication mechanism designed to establish secure communication between a user





**Figure 11:** Illustration of the user interface showcasing successful login and authentication in the implemented system



**Figure 12:** Visualization of the Session Key Generation Process Between User and Server

and a server in a database-driven environment. The proposed protocol ensures mutual authentication, enabling both entities to validate each other's identity before engaging in a secure session. Additionally, a session key is dynamically generated to safeguard subsequent interactions, preserving data confidentiality and integrity. The implementation, developed in Python 3.9, leverages cryptographic techniques to mitigate unauthorized access and security threats effectively. To rigorously evaluate the protocol's security, formal validation was conducted using the Real-or-Random (RoR) model. Furthermore, automated security verification was performed through ProVerif and Scyther to assess its resilience against various attack scenarios. The analysis confirms that the pro-

posed authentication scheme successfully mitigates critical security threats, including impersonation, replay, and key compromise impersonation attacks, while demonstrating improved performance in terms of computational efficiency, communication overhead, and security robustness compared to existing authentication mechanisms. While the proposed protocol exhibits strong security properties, certain challenges remain, particularly in terms of scalability and interoperability with existing database infrastructures. To further validate its effectiveness, the system was implemented and tested under controlled conditions, assessing its practical feasibility and operational performance. This authentication protocol is particularly suitable for environ-

**Table 3**  
Execution Details of the Implemented Authentication Protocol

Parameter	Hexadecimal Value
$ID_i$	0x75736572313233 (hex for "user123")
$PW_i$	0x70617373776f7264313233 (hex for "password123")
$K_i$	0x6b6579313233 (hex for "key123")
$HID_i$	0x9a8f4b2c76e0d1c854a67f9b3e25d8a1f4c2e5b7d6a3e9c0f7b4d8a2c5e3f1b0
$HPW_i$	0xb1c4d7e8a9f23b65e4c1f0a7d95e3b2c4d7a1e9c6f8b3e2d5c7a9f04e1b2d3c
$K_j$	0x6b65796a313233 (hex for "keyj123")
$RID_i$	0xc8a5f4d2b6e3c1d7a9f8e2b3d5c7a1e9f0b4d8a2c5e7b0d3a9f1c4e6b2d8a7f0
$SB_i$	0xd3b2a5e7f0c4d8b9f1c7e5a3d2b6f8e1c9d4a0b7e3c5a8f2d9b1e6c0f4a7d8e3b5
$A_i$	0xf4d2a9b7e3c5a8f1c7d5e2b6f0a3d9c1b8e7c4a5f2d3b0e9c6a7f1d8b2e4c9a5f3
$B_i$	0xb7c5a8f2d9e1b3d6a0f4c7e5a9b2d8e3c1f0a7b5e2c9d3a6f1b4e0c8d7a5f2b9e6
$C_i$	0xe9d3b0a5f2c8b7e4a6d9c1f0b2d8e3c5a7f1d6b4e0c9a8f3d2b5e1c7a9f0d8b6e2
$D_i$	0xa5f1d8b3e6c9a7f2d4b0e9c8a6f3d2b5e1c7a9f0d8b6e4c1a9f2d5b3e0c8a7f1d6
$USN_i$	0xc4b8f1d7a5e2b9d6c1f0a3d8e5b2c7a9f4d3b0e7c6a8f2d5b1e9c0a7f3d4b6e1
$UF_i$	0xd9b5a7f2c8e1b3d6a0f4c7e5a9b2d8e3c1f0a7b5e2c9d3a6f1b4e0c8d7a5f2b9e6
$T_1$	0x74696d657374616d705f313233 (hex for "timestamp_123")
$T_2$	0x74696d657374616d705f313234 (hex for "timestamp_124")
$Sk$	0xf1b7a5e2c8d9b3e6c1f0a7d5b2e4c9a6f3d8b0e7c5a9f2d4b1e9c0a8f3d7b6e1c4
$Sk^*$	0xf1b7a5e2c8d9b3e6c1f0a7d5b2e4c9a6f3d8b0e7c5a9f2d4b1e9c0a8f3d7b6e1c4 Verification? True ( $Sk^* = Sk$ )

**Table 4**  
Security Comparison of the Proposed Protocol with Recent Authentication Schemes

Protocols	$A_1$	$A_2$	$A_3$	$A_4$	$A_5$	$A_6$	$A_7$
Aghili et al. (2019b)	✓	✗	✓	✓	✗	✓	✓
Li et al. (2019)	✓	✗	✓	✗	✓	✓	✓
Fotouhi et al. (2020)	✓	✓	✗	✗	✓	✓	✗
Amintoosi et al. (2022)	✗	✓	✓	✓	✓	✓	✗
Fariss et al. (2022)	✓	✗	✓	✓	✓	✗	✗
Wu et al. (2023)	✗	✓	✗	✗	✗	✓	✓
Ali and Ahmed (2024)	✓	✓	✗	✓	✓	✓	✓
LAPWN	✓	✓	✓	✓	✓	✓	✓

$A_1$ : Protection against replay attacks;  $A_2$ : Defense against privileged insider threats;  
 $A_3$ : Prevention of session key disclosure;  $A_4$ : Ensuring user untraceability;  
 $A_5$ : Resistance to spoofing attacks;  $A_6$ : Guaranteeing forward secrecy;  
 $A_7$ : Mitigation of de-synchronization attacks;  
 ✓: Strong security feature    ✗: Lacks security feature

ments requiring high security, such as user-server authentication in medical record management, e-government services, and online banking systems. Future research will focus on integrating the protocol with advanced security technologies, such as blockchain for decentralized authentication, artificial intelligence (AI) for adaptive security mecha-

nisms, and enhanced cryptographic functions for improved resilience. Additionally, further evaluations will be conducted in various real-world computing environments to assess usability, efficiency, and security under diverse operational conditions, ensuring its adaptability to broader applications.

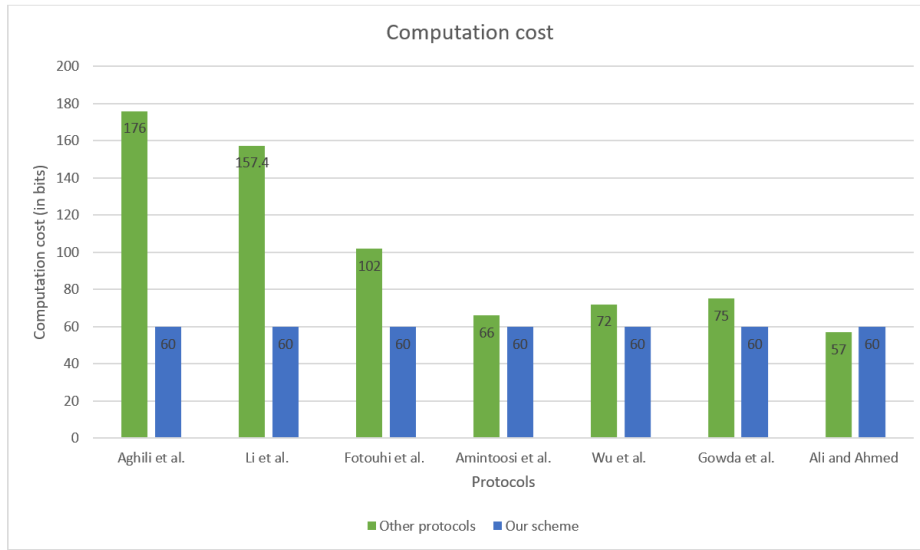
**Table 5**  
Notation and Metrics for Evaluating Computational and Communication Overhead

Symbol	Definition	Execution Time	Data Size	Transmission
$T_h$	Processing duration required for performing a cryptographic hash function	3ms	256 bits	
$T_{en/d}$	Time taken to execute both encryption and decryption operations	3.7ms	128 bits	
$T_{tc}$	Size of the timestamp utilized in message exchanges	-	32 bits	

**Table 6**

Computational Cost Comparison of the Proposed Authentication Protocol with Recent Schemes (in Milliseconds)

Protocols	Overall Computational Cost for $S_j$ , and $U_i$ in ms
Aghili et al. (2019b)	$(27 + 3m)T_h + 1T_{bh} = 176$ ms where $m = 10$ is the number of sensors
Li et al. (2019)	$22T_h + 4T_{mu} + T_{E/D} = 157.4$ ms
Fotouhi et al. (2020)	$32T_h = 102$ ms
Amintoosi et al. (2022)	$22T_h = 66$ ms
Wu et al. (2023)	$24T_h = 72$ ms
Gowda et al. (2024)	$25T_h = 75$ ms
Ali and Ahmed (2024)	$19T_h = 57$ ms
LAPWN	$20T_h = 60$ ms



**Figure 13:** A Comprehensive Comparison of the Total Computational Overhead of the Proposed Authentication Protocol with Recent Authentication Schemes

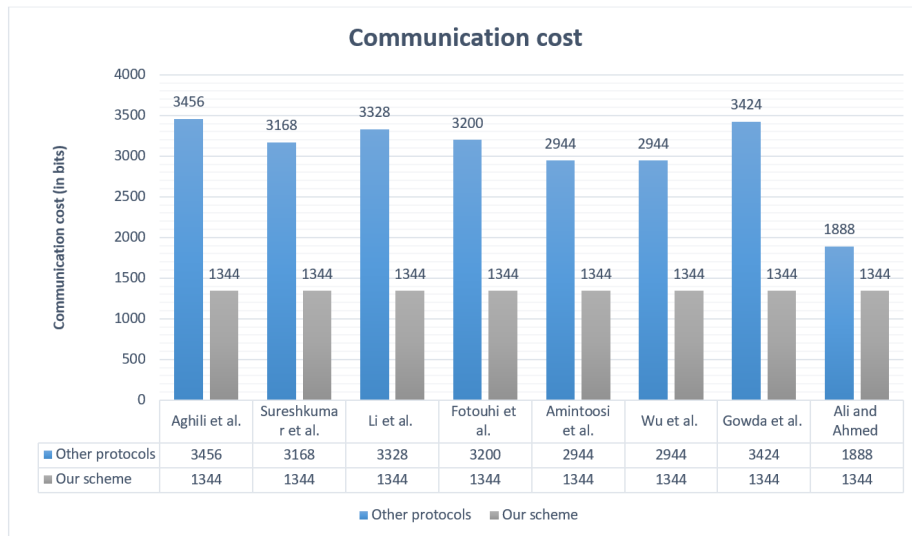
## References

- Aghili, S.F., Mala, H., Kaliyar, P., Conti, M., 2019a. Seclap: Secure and lightweight rfid authentication protocol for medical iot. *Future Generation Computer Systems* 101, 621–634.
- Aghili, S.F., Mala, H., Shojafar, M., Peris-Lopez, P., 2019b. Laco: Lightweight three-factor authentication, access control and ownership transfer scheme for e-health systems in iot. *future generation computer systems* 96, 410–424.
- Ali, H., Ahmed, I., 2024. Laaka: Lightweight anonymous authentication and key agreement scheme for secure fog-driven iot systems. *Computers Security* 140, 103770. URL: <https://www.sciencedirect.com/science/article/pii/S0167404824000713>, doi:<https://doi.org/10.1016/j.cose.2024.103770>.
- Alzahrani, B.A., Irshad, A., Albeshri, A., Alsubhi, K., 2021. A provably secure and lightweight patient-healthcare authentication protocol in wireless body area networks. *Wireless Personal Communications* 117, 47–69.
- Amin, R., Islam, S.H., Biswas, G., Khan, M.K., Kumar, N., 2018. A robust and anonymous patient monitoring system using wireless medical sensor

**Table 7**

Comparative Analysis of Communication Overhead in the Proposed Authentication Protocol Versus Recent Authentication Schemes

Protocols	Year	Overall Communication Cost for $S_j$ and $U_i$ (in bits)
Aghili et al. (2019b)	2019	3456
Li et al. (2019)	2019	3328
Fotouhi et al. (2020)	2020	3200
Amintoosi et al. (2022)	2022	2944
Wu et al. (2023)	2023	2944
Gowda et al. (2024)	2024	3424
Ali and Ahmed (2024)	2024	1888
LAPWN	-	1344



**Figure 14:** Comparison of the overall communication overhead of the proposed authentication protocol with recent schemes

- networks. *Future Generation Computer Systems* 80, 483–495.
- Amintoosi, H., Nikooghadam, M., Shojafar, M., Kumari, S., Alazab, M., 2022. Slight: A lightweight authentication scheme for smart healthcare services. *Computers and Electrical Engineering* 99, 107803.
- Arslan, A., Bingöl, M.A., 2022. Security and privacy analysis of recently proposed ecc-based rfid authentication schemes. *Cryptology ePrint Archive*.
- Challa, S., Das, A.K., Odelu, V., Kumar, N., Kumari, S., Khan, M.K., Vasilakos, A.V., 2018. An efficient ecc-based provably secure three-factor user authentication and key agreement protocol for wireless healthcare sensor networks. *Computers & Electrical Engineering* 69, 534–554.
- Chander, B., Gopalakrishnan, K., 2022. A secured and lightweight rfid-tag based authentication protocol with privacy-preserving in telecare medicine information system. *Computer Communications* 191, 425–437.
- Chandrakar, P., Om, H., 2018. An extended ecc-based anonymity-preserving 3-factor remote authentication scheme usable in tmis. *International Journal of Communication Systems* 31, e3540.
- Chatterjee, U., Ray, S., Adhikari, S., Khan, M.K., Dasgupta, M., 2023. An improved authentication and key management scheme in context of iot-based wireless sensor network using ecc. *Computer Communications* 209, 47–62.
- Chen, Y., Zhang, N., Zhang, Y., Chen, X., Wu, W., Shen, X., 2019. Energy efficient dynamic offloading in mobile edge computing for internet of things. *IEEE Transactions on Cloud Computing* 9, 1050–1060.
- Das, A.K., Wazid, M., Kumar, N., Vasilakos, A.V., Rodrigues, J.J., 2018. Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment. *IEEE Internet of Things Journal* 5, 4900–4913.
- Dolev, D., Yao, A., 1983. On the security of public key protocols. *IEEE Transactions on information theory* 29, 198–208.
- Fariss, M., El Gafif, H., Toumanari, A., 2022. A lightweight ecc-based three-factor mutual authentication and key agreement protocol for wsns in iot. *International Journal of Advanced Computer Science and Applications* 13.
- Fotouhi, M., Bayat, M., Das, A.K., Far, H.A.N., Pournaghi, S.M., Doostari, M.A., 2020. A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care iot. *Computer Networks* 177, 107333.
- Gabsi, S., Kortli, Y., Beroulle, V., Kieffer, Y., Alasiry, A., Hamdi, B., 2021. Novel ecc-based rfid mutual authentication protocol for emerging iot applications. *IEEE Access* 9, 130895–130913.
- Gowda, N.C., Manvi, S.S., Malakreddy, A.B., Buyya, R., 2024. Takm-fc: Two-way authentication with efficient key management in fog computing environments. *The Journal of Supercomputing* 80, 6855–6890.
- Gupta, A., Tripathi, M., Shaikh, T.J., Sharma, A., 2019. A lightweight anonymous user authentication and key establishment scheme for wearable devices. *Computer Networks* 149, 29–42.
- Hajian, R., ZakeriKia, S., Erfani, S.H., Mirabi, M., 2020. Shaparak: Scalable healthcare authentication protocol with attack-resilience and anonymous key-agreement. *Computer Networks* 183, 107567.
- Hosseinizadeh, M., Servati, M.R., Rahmani, A.M., Safkhani, M., Lansky, J., Janoscova, R., Ahmed, O.H., Tanveer, J., Lee, S.W., 2024. An enhanced authentication protocol suitable for constrained rfid systems. *IEEE Access*.
- Huang, W., 2024. Ecc-based three-factor authentication and key agreement scheme for wireless sensor networks. *Scientific Reports* 14, 1787.
- Jegadeesan, S., Azees, M., Rajasekaran, A.S., Al-Turjman, F., 2022. Lightweight privacy and confidentiality preserving anonymous authentication scheme for wbans. *IEEE Trans. Ind. Informatics* 18, 3484–3491. URL: <https://doi.org/10.1109/TII.2021.3097759>, doi:10.1109/TII.2021.3097759.
- Jia, X., Luo, M., Wang, H., Shen, J., He, D., 2022. A blockchain-assisted privacy-aware authentication scheme for internet of medical things. *IEEE Internet of Things Journal* 9, 21838–21850.
- Khan, M.A., Ullah, S., Ahmad, T., Jawad, K., Buriro, A., 2023a. Enhancing security and privacy in healthcare systems using a lightweight RFID protocol. *Sensors* 23, 5518.
- Khan, N., Zhang, J., Mallah, G.A., Chaudhry, S.A., 2023b. A secure and efficient information authentication scheme for e-healthcare system. *Computers, Materials & Continua* 76.
- Kumar, P., Pal, A.K., Islam, S.H., 2024. 2f-mask-vss: Two-factor mutual authentication and session key agreement scheme for video surveillance system. *Journal of Systems Architecture*, 103196.
- Kumar, V., Ahmad, M., Kumari, A., 2019. A secure elliptic curve cryptography based mutual authentication protocol for cloud-assisted tmis. *Teleomatics and Informatics* 38, 100–117.
- Li, C.T., Shih, D.H., Wang, C.C., 2018. Cloud-assisted mutual authentication and privacy preservation protocol for telecare medical information systems. *Computer methods and programs in biomedicine* 157, 191–203.
- Li, X., Peng, J., Obaidat, M.S., Wu, F., Khan, M.K., Chen, C., 2019. A secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems. *IEEE Systems Journal* 14, 39–50.
- Oh, J., Yu, S., Lee, J., Son, S., Kim, M., Park, Y., 2021. A secure and



- lightweight authentication protocol for iot-based smart homes. *Sensors* 21, 1488.
- Pu, C., Zerkle, H., Wall, A., Lim, S., Choo, K.K.R., Ahmed, I., 2022. A lightweight and anonymous authentication and key agreement protocol for wireless body area networks. *IEEE Internet of Things Journal* 9, 21136–21146.
- Rani, D., Tripathi, S., 2024. Design of blockchain-based authentication and key agreement protocol for health data sharing in cooperative hospital network. *The Journal of Supercomputing* 80, 2681–2717.
- Sadhukhan, D., Ray, S., Biswas, G., Khan, M.K., Dasgupta, M., 2021. A lightweight remote user authentication scheme for iot communication using elliptic curve cryptography. *The Journal of Supercomputing* 77, 1114–1151.
- Saeed, M.E.S., Liu, Q.Y., Tian, G., Gao, B., Li, F., 2018. Remote authentication schemes for wireless body area networks based on the internet of things. *IEEE Internet of Things Journal* 5, 4926–4944. doi:10.1109/JIOT.2018.2876133.
- Safkhani, M., Rostampour, S., Bendavid, Y., Bagheri, N., 2020. Iot in medical & pharmaceutical: Designing lightweight rfid security protocols for ensuring supply chain integrity. *Computer Networks* 181, 107558.
- Safkhani, M., Vasilakos, A., 2019. A new secure authentication protocol for telecare medicine information system and smart campus. *IEEE Access* 7, 23514–23526.
- Servati, M.R., Safkhani, M., 2023. Eccbas: An ecc based authentication scheme for healthcare iot systems. *Pervasive and Mobile Computing*, 101753 URL: <https://www.sciencedirect.com/science/article/pii/S1574119223000111>, doi:<https://doi.org/10.1016/j.pmcj.2023.101753>.
- Servati, M.R., Safkhani, M., Ali, S., Malik, M.H., Ahmed, O.H., Hosseinzadeh, M., Mosavi, A.H., 2022. Cryptanalysis of two recent ultralightweight authentication protocols. *Mathematics* 10, 4611.
- Servati, M.R., Safkhani, M., Rahmani, A.M., Hosseinzadeh, M., 2025. Erasmus: An ecc-based robust authentication protocol suitable for medical iot systems. *Computer Networks* 258, 110938. URL: <https://www.sciencedirect.com/science/article/pii/S1389128624007709>, doi:<https://doi.org/10.1016/j.comnet.2024.110938>.
- Shariq, M., Singh, K., Maurya, P.K., Ahmadian, A., Ariffin, M.R.K., 2021. URASP: an ultralightweight RFID authentication scheme using permutation operation. *Peer-to-Peer Networking and Applications* 14, 3737–3757.
- Shuai, M., Yu, N., Wang, H., Xiong, L., 2019. Anonymous authentication scheme for smart home environment with provable security. *Computers & Security* 86, 132–146.
- Suganthi, S., Anitha, R., Sureshkumar, V., Harish, S., Agalya, S., 2020. End to end light weight mutual authentication scheme in iot-based healthcare environment. *Journal of Reliable Intelligent Environments* 6, 3–13.
- Sureshkumar, V., Amin, R., Vijaykumar, V., Sekar, S.R., 2019. Robust secure communication protocol for smart healthcare system with fpga implementation. *Future Generation Computer Systems* 100, 938–951.
- Wang, W., Yan, B., Chai, B., Shen, R., Dong, A., Yu, J., 2024. Ebias: Ecc-enabled blockchain-based identity authentication scheme for iot device. *High-Confidence Computing*, 100240.
- Wang, X., Fan, K., Yang, K., Cheng, X., Dong, Q., Li, H., Yang, Y., 2022. A new rfid ultra-lightweight authentication protocol for medical privacy protection in smart living. *Computer Communications* 186, 121–132.
- Wang, Y., Liu, Y., 2022. Rc2pas: Revocable certificateless conditional privacy-preserving authentication scheme in wbans. *IEEE Systems Journal* 16, 5675–5685.
- Wu, T.Y., Wang, L., Chen, C.M., 2023. Enhancing the security: A lightweight authentication and key agreement protocol for smart medical services in the ioht. *Mathematics* 11. URL: <https://www.mdpi.com/2227-7390/11/17/3701>, doi:10.3390/math11173701.
- Xiang, A., Zheng, J., 2020. A situation-aware scheme for efficient device authentication in smart grid-enabled home area networks. *Electronics* 9, 989.
- Xu, Z., Xu, C., Chen, H., Yang, F., 2019. A lightweight anonymous mutual authentication and key agreement scheme for wban. *Concurrency and computation: Practice and experience* 31, e5295.
- Yang, X., Yi, X., Nepal, S., Khalil, I., Huang, X., Shen, J., 2022. Efficient and anonymous authentication for healthcare service with cloud based wbans. *IEEE Transactions on Services Computing* 15, 2728–2741. doi:10.1109/TSC.2021.3059856.
- Zheng, L., Song, C., Cao, N., Li, Z., Zhou, W., Chen, J., Meng, L., 2018. A new mutual authentication protocol in mobile rfid for smart campus. *IEEE Access* 6, 60996–61005.