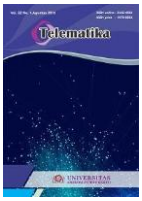

Available online at :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

Telematika

Accreditated SINTA "2" Kemdiktisaintek, No.10/C/C3/DT.05.00/2025

UNIVERSITAS AMIKOM PURWOKERTO

Cryptanalysis of Two-Factor Authentication in the Internet of Vehicles Network Environment

E Haodudin Nurkifli[1,*]

[1]Department of Informatics, Faculty of Computer Science, Universitas Singaperbangsa Karawang, Karawang, Indonesia

ARTICLE INFO

History of the article:
Received June 26, 2026
Revised August 3, 2026
Accepted August 5, 2026

Keywords:
Authentication
Cryptanalysis
Protocol Security

Correspondence:
dudi.nurkifli@staff.unsika.ac.id

ABSTRACT

An Intelligent Transportation System (ITS) plays a vital role in smart city ecosystems by enabling vehicles to communicate with roadside units, cloud servers, and other vehicles through the Internet. Numerous authentication protocols have been proposed to secure communications in the Internet of Vehicles (IoV), and recent studies have focused on enhancing user privacy through two-factor authentication. However, a detailed cryptanalysis conducted in this study reveals that an existing authentication protocol still suffers from several security vulnerabilities. Furthermore, under a stronger attacker model, an adversary who physically captures a device may extract the stored credentials, impersonate a legitimate user, and clone the device. To address these limitations, this study first presents a cryptanalysis of the existing protocol and then proposes a new authentication protocol. The proposed protocol integrates biometric authentication with a fuzzy extractor to derive cryptographic keys from fingerprint data, thereby preventing attackers from obtaining valid secret credentials even if a device is physically captured. Formal security analysis based on the Real-or-Random (RoR) model demonstrates that the proposed authentication protocol achieves strong anonymity, mutual authentication, resilience against desynchronization, and perfect forward and backward secrecy while resisting replay, impersonation, denial-of-service (DoS), and physical attacks. Furthermore, formal verification using the Scyther tool confirms that the proposed authentication protocol satisfies all specified security claims without identifying any potential attacks. Performance evaluation demonstrates that the proposed authentication protocol achieves the lowest execution time among the compared protocols, indicating its suitability for resource-constrained IoV devices.

1. INTRODUCTION

In Intelligent Transportation Systems (ITS) for smart cities, communication technologies continue to evolve to support interactions between vehicles and their surrounding environments. These include Vehicle-to-Vehicle (V2V), Vehicle-to-Roadside (V2R), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Everything (V2X) communication (Shi et al., 2025), (Aman & Javaid, 2021), (Xue et al., 2022). V2V enables vehicles to exchange information with one another to detect nearby vehicles and prevent road accidents. V2R refers to communication between vehicles and roadside units positioned along the road, which typically forward relevant information to the service provider. V2I supports communication between vehicles and infrastructure components such as traffic lights, road signs, and cameras, allowing vehicles to receive timely warnings about road conditions or accidents ahead. V2X integrates all these communication

types, enabling vehicles to interact with diverse entities in the ITS environment (Yafoz et al., 2026), (Singh et al., 2026), (Khan et al., 2026).

Because these communication channels rely on wireless transmission over public networks, they are exposed to significant security threats such as eavesdropping, replay attacks, and denial-of-service (DoS) attacks. A particularly severe threat occurs when an attacker physically steals a vehicle's onboard device, especially when the vehicle is parked in a public area. Similarly, attackers may steal a smart card used for vehicle access, enabling them to unlock and operate the vehicle illegally.

Several researchers have attempted to address the security challenges in the Internet of Vehicles (IoV), which serves as the communication backbone of Intelligent Transportation Systems (ITS) (Zhang et al., 2023), (Javaid & Aman, 2020), (J. Wang et al., 2022), (Zhang et al., 2023). Among them, (Sibahee et al., 2024) proposed a two-factor authentication protocol to enhance security in IoV. However, the cryptanalysis presented in this study reveals that their protocol contains several security weaknesses and remains vulnerable to well-known attacks.

Recent studies have proposed various authentication protocols to improve IoV security. (Tiwari & Kumar, 2026) and (Huang & Chen, 2026) proposed blockchain-based authentication protocols for the IoV environment. However, their protocols do not provide resistance against physical attacks and fail to address the desynchronization problem caused by communication flooding attacks. (S. Kumar et al., 2026) proposed an ultra-lightweight authentication protocol; however, it does not provide resistance to denial-of-service (DoS) attacks or strong anonymity. Similarly, (Lin & Yang, 2026) proposed a PUF-based authentication protocol. Nevertheless, their protocol does not provide resistance against physical attacks because an attacker who captures a device may compromise the stored credentials or exploit the programmable characteristics of the PUF. Motivated by these limitations, this study proposes a new authentication protocol to address the identified weaknesses and performs formal security analysis to demonstrate that the proposed authentication protocol satisfies the required security properties and resists various security attacks. The main contributions of this study are summarized as follows.

1) Cryptanalysis of Sibahee et al.'s protocol to identify its loopholes and security vulnerabilities. (Sibahee et al., 2024)
2) Proposal of a new authentication protocol based on biometrics and fuzzy extractors. The use of biometric-based fuzzy extractors helps address noise in fingerprint readings and enhances resistance to side-channel attacks and device-theft scenarios. The proposed protocol also preserves essential security features such as mutual authentication, anonymity, and others.
3) Formal security analysis using the Real-or-Random (RoR) model, which simulates realistic attack scenarios such as device theft, smart-card extraction, and replay attacks. The results demonstrate that the proposed protocol satisfies the expected security requirements under the RoR model.

The remainder of this manuscript is organized as follows. Section I presents the introduction. Section II describes the preliminaries and fundamental concepts. Section III provides the methodology. Section IV reviews Sibahee's protocol, presents its cryptanalysis, introduces the proposed protocol, and discusses the formal analysis. Finally, Section V concludes the paper.

2. RESEARCH METHODS

This section explains about the materials of the research and the equipments used as the method. There is also explanations about the steps of solving the problem.

2.1. Biometrics

Biometrics is one of the authentication factors that leverages unique human characteristics, such as fingerprints. During the registration phase, the user provides a fingerprint (Bio), which is stored in processed form on the device. During login, the user presents a new fingerprint sample (Bio'), and the system compares Bio and Bio'. If they match, authentication is successful. However, biometric samples are rarely identical over time due to noise, environmental factors, sensor imperfections, or changes in the user's finger condition. As a result, Bio and Bio' may differ slightly, causing authentication failures. To overcome this issue, a fuzzy extractor is employed to stabilize the biometric features and tolerate the inherent noise (Sibahee et al., 2024), (Zahednejad & Gao, 2023).

2.2. Fuzzy Extractor (FE)

A fuzzy extractor consists of two core functions: Generate and Reconstruct. Generate: This function takes a biometric input Bio and outputs a stable secret key along with helper data: $(KeyBio, HelpData) = Generate(Bio)$. The helper data does not reveal the biometric information but enables successful reconstruction later. Reconstruct: This function takes a noisy biometric sample Bio' along with the helper data and reconstructs the same secret key as long as Bio' is sufficiently close to the original Bio: $KeyBio = Reconstruct(Bio', HelpData)$. The fuzzy extractor thus ensures that even when the biometric input varies slightly, a consistent and reliable key can still be generated for authentication (Dodis et al., 2004), (X. Wang & Xie, 2025), (Wen & Su, 2025), .

2.3. Attacker model

The attacker model considered in this study follows the Dolev–Yao (DY) model (Dolev & Yao, 1983) and (N. Kumar & Chaudhary, 2026). Under this model, the attacker has full control over the public communication channel and can intercept, eavesdrop, replay, modify, inject, and block transmitted messages. Furthermore, the attacker is assumed to possess stronger capabilities, including physically capturing a device and extracting the credentials stored in its memory.

2.4. Security Requirements for the IoV Environment

1) Strong Anonymity: The protocol preserves the anonymity of legitimate users by preventing the disclosure of their real identities while ensuring untraceability across multiple communication sessions.
2) Perfect Forward and Backward Secrecy: The protocol ensures that the compromise of current secret credentials does not affect the confidentiality of previously established or future session keys.
3) Mutual Authentication: The protocol enables the communicating parties to authenticate each other before establishing a secure communication session.

4) Resilience Against Desynchronization: The protocol incorporates a recovery mechanism to maintain synchronization between communicating entities in the presence of communication interruptions caused by denial-of-service (DoS) attacks, network failures, or power outages.
5) Resistance to Physical Attacks: The protocol protects sensitive credentials even if an attacker physically captures a device and attempts to extract the information stored in its memory.

2.5. Method

The methodology of this research consists of several key steps, described as follows:

1) In-depth analysis of Sibahee et al.'s protocol. A comprehensive review is conducted to fully understand the structure, operations, and security assumptions of the protocol proposed by Sibahee et al.
2) Cryptanalysis. The original protocol is reconstructed, and multiple attack scenarios are applied to identify potential loopholes and security weaknesses. This step evaluates how the protocol behaves under common adversarial conditions.
3) Proposal of a new protocol. After identifying the vulnerabilities in Sibahee et al.'s scheme, a new and improved authentication protocol is designed. In this study, the proposed protocol incorporates biometric authentication combined with a fuzzy extractor to enhance robustness and resist security threats such as device theft and side-channel attacks.
4) Formal security analysis. The Real-or-Random (RoR) model is used to formally verify that the proposed authentication protocol achieves the required security properties. This analysis includes mathematical formulations and security proofs to demonstrate resistance against well-known attacks, including replay, impersonation, and device compromise attacks. In addition, formal security verification is performed using the Scyther tool to provide further evidence that the proposed authentication protocol satisfies the specified security claims.

3. RESULTS AND DISCUSSION

This Section consists of a brief review of Sibahee et al.'s protocol, the cryptanalysis of their scheme, the proposed new protocol, and the formal analysis using the RoR model.

3.1. Review of the Sibahee et al.'s protocol

This subsection presents a brief overview of the protocol designed by Sibahee et al., which consists of two main phases: registration and mutual authentication.

3.1.1. Registration between RSU and Server

The server selects the identity ID_{R_j} and the key K_{T_A}, computes $K_{TA-Rj} = h\left\{ID_{R_j} || K_{T_A}\right\}$, stores ID_{R_j}, and sends ID_{R_j}, K_{TA-Rj} to the RSU. Upon receiving the values, the RSU stores the received information.

3.1.2. Registration between Onboard Unit (OBU)/User and Server

Step 1: The OBU selects ID_{U_i}, PW, η_1 and computes: $A_1 = h(ID_{U_i}||PW||\eta_1)$, $A_2 = h(ID_{U_i}||\eta_1)$, then constructs the registration request: $\{ID_{U_i}, \eta_1, A_1, A_2\}$and sends it to the server. Step 2: Upon receiving the registration request, the server verifies whether $A_2 = h(ID_{U_i}||\eta_1)$ holds. It stores A_2, and computes:

$A_3 = h(ID_{TA}||K_{T_A}||A_2) \oplus A_1$, $A_4 = h(A_2||K_{T_A}) \oplus h(A_1||A_2)$, $A_5 = h(A_1||A_2||A_4)$. The server constructs the registration response $\{A_3, A_4, A_5, ID_{TA}\}$ and sends it to the OBU. Step 3: Upon receiving the response, the OBU computes $B_1 = h(ID_{U_i}||PW) \oplus \eta_1$ and stores $\{A_3, A_4, A_5, B_1, ID_{TA}\}$.

3.1.3. Mutual Authentication

This subsection describes the mutual authentication procedure defined in the Sibahee et al. protocol.

Step 1: Authentication Initiation (OBU → Server). The user inputs (ID_{U_i}, PW). The OBU derives: $\eta_1 = h(ID_{U_i}||PW) \oplus B_1$, $A_1 = h(ID_{U_i}||PW||\eta_1)$, $A_2 = h(ID_{U_i}||\eta_1)$. It verifies: A_5=? $h(A_1||A_2||A_4)$. Then the OBU generates η_2 and computes: $B_2 = A_3 \oplus A_1 = h(ID_{TA}||K_{T_A}||A_2)$, $B_3 = B_2 \oplus \eta_2$, $B_4 = h(ID_{TA}||ID_{R_j}||B_2||A_2||\eta_2||T_1)$. The OBU constructs: $L_M = \{B_3, B_4, ID_{TA}, A_2, ID_{R_j}, T_1\}$ and sends it to the server.

Step 2: Server Verification and Response (Server → RSU). Upon receiving L_M, the server generates T_2, verifies ID_{TA} & T_1, and computes: $B_2 = A_3 \oplus A_1 = h(ID_{TA}||K_{T_A}||A_2)$, $\eta_2^* = B_2 \oplus B_3$, $B_4^* = h(ID_{TA}||ID_{R_j}||B_2||A_2||\eta_2^*||T_1)$. If $B_4^* = B_4$, the server retrieves ID_{R_j}, generates η_3 and T_3, and computes: $K_{TA-Rj} = h\{ID_{R_j}||K_{T_A}\}$, $B_5 = h(K_{TA-Rj}||ID_{R_j}||ID_{TA}) \oplus \eta_2$, $C_1 = h(\eta_2) \oplus \eta_3$, $C_2 = h(K_{TA-Rj}||\eta_2||\eta_3||T_3)$. The server prepares: $AUT_1 = \{ID_{R_j}, B_5, C_1, C_2, T_3\}$ and sends it to the RSU.

Step 3: RSU Verification and Response (RSU → Server). Upon receiving AUT_1, the RSU computes T_4, ID_{R_j} & T_3, $\eta_2^* = h(K_{TA-Rj}||ID_{R_j}||ID_{TA}) \oplus B_5$, $\eta_3^* = h(\eta_2) \oplus C_1$, $C_2^* = h(K_{TA-Rj}||\eta_2^*||\eta_3^*||T_3)$. If $C_2^* = C_2$, , it generates η_4 and computes: $SK_R = h(\eta_2||\eta_3||\eta_4)$, $C_3 = h(K_{TA-Rj}||\eta_3) \oplus \eta_4$, $C_4 = ID_{R_j} \oplus h(\eta_4||C_3)$, $C_5 = h(SK_R||ID_{TA}||ID_{R_j}||\eta_4||T_4)$. The RSU constructs: $AUT_2 = \{C_3, C_4, C_5, T_4\}$ and sends it to the server.

Step 4: Server Verification and Response (Server → OBU). Upon receiving AUT_2, the server generates T_5, verifies T_4, and computes: $\eta_4^* = h(K_{TA-Rj}||\eta_3) \oplus C_3$, $ID_{R_j}^* = C_4 \oplus h(\eta_4^*||C_3)$. The server verifies whether $ID_{R_j}^*$ =? ID_{R_j}, then computes: $C_5^* = h(SK_R||ID_{TA}||ID_{R_j}^*||\eta_4^*||T_4)$, and verifies C_5^* =? C_5. It computes: $SK_T = h(\eta_2||\eta_3||\eta_4)$, $D_1 = h(\eta_2||A_2)$, $D_2 = h(\eta_2||\eta_3) \oplus \eta_4$, $D_3 = h(SK_T||A_2||\eta_3||\eta_4||T_5)$, $AUT_3 = \{D_1, D_2, D_3, T_5\}$. The server sends: $AUT_3 = \{D_1, D_2, D_3, T_5\}$ to the OBU.

Step 5: Final Verification (OBU). Upon receiving AUT_3, the OBU generates T_6, verifies T_5, and computes: $\eta_3^* = D_1 \oplus h(\eta_2||A_2)$, $\eta_4^* = D_2 \oplus h(\eta_2||\eta_3)$, $SK_B = h(\eta_2||\eta_3^*||\eta_4^*)$, $D_3^* = h(SK_B||A_2||\eta_3^*||\eta_4^*||T_5))$. It verifies: D_3^* =? D_3. If valid, the shared session keys match: $SK_B = SK_T = SK_R$.

3.2. Cryptanalysis of the Sibahee et al.'s protocol

This subsection presents the weaknesses identified in the protocol proposed by Sibahee et al. The vulnerabilities are summarized and explained as follows.

3.2.1. Inability to Withstand Side-Channel Attacks

The protocol fails to resist side-channel attacks. The evidence is shown through the following adversarial scenario:

1) The attacker physically steals the user's device.
2) The attacker obtains all transmitted data from the public communication channel.
3) The attacker performs a side-channel attack on the stolen device.
4) Because the device stores $\{A_3, A_4, A_5, B_1, ID_{TA}\}$ in plaintext and not in encrypted form, the attacker can easily extract these values.

Based on this scenario, it is clear that the protocol does **not** provide proper protection against side-channel attacks.

3.2.2. Vulnerability to Password and Identity Guessing Attacks

The protocol also fails to withstand password-guessing and identity-guessing attacks. The attack steps are as follows:

1) The attacker guesses a password candidate PW_i^* from the password dictionary D_{pw}.
2) The attacker guesses an identity candidate $ID_{U_i}^*$ from the identity dictionary D_{id}.
3) The attacker computes: $\eta_1^* = h\left(ID_{U_i}^* || PW_i^*\right) \oplus B_1$.
4) The attacker computes: $A_1^* = h\left(ID_{U_i}^* || PPW_i^* || \eta_1^*\right), A_2^* = h\left(ID_{U_i}^* || \eta_1^*\right), A_5^* = h(A_1^* || A_2^* || A_4)$.
5) If $A_5^* = A_5$, the attacker has discovered the correct identity and password pair.

Once the attacker obtains valid credentials, they can perform *impersonation attacks* and *other credential-based attacks*. The above steps clearly show that the protocol cannot withstand password-guessing or identity-guessing attacks.

3.2.3. Vulnerability to Impersonation Attacks

The protocol also allows an attacker to impersonate a legitimate user. The attack proceeds as follows:

1) The attacker inputs the guessed identity $ID_{U_i}^*$ and password PW_i^* into the device.
2) The attacker computes: $\eta_1^* = h\left(ID_{U_i}^* || PW_i^*\right) \oplus B_1$, $A_1^* = h\left(ID_{U_i}^* || PW_i^* || \eta_1^*\right), A_2^* = h\left(ID_{U_i}^* || \eta_1^*\right)$, $A_5^* = h(A_1^* || A_2^* || A_4)$.
3) If the verification $A_5^* = A_5$ succeeds, this confirms that the attacker's guessed credentials are correct.

Thus, the attacker can authenticate as the legitimate user. This demonstrates that the protocol is vulnerable to impersonation attacks.

3.2.4. Failure to Achieve User Anonymity

The protocol does not protect user identity. Because an attacker can recover the actual identity ID_{U_i} through eavesdropping, message interception, or by reading unprotected values stored in the device, the protocol fails to provide identity protection. Therefore, user anonymity is not achieved.

3.2.5. Lack of Perfect Forward and Backward Secrecy

The protocol does not update critical values or long-term secrets in each session. As a result: If an attacker obtains the current session key, they can derive both **past** and **future** session keys. This violates the security requirements of: Perfect Forward Secrecy (PFS) and Backward Secrecy (BFS). The absence of session-dependent key updates makes the protocol weak against key-compromise attacks.

3.3. Proposed protocol

This subsection presents the proposed authentication protocol, which consists of two main processes: registration and mutual authentication between the vehicle (user/smartcard) and the server. Before discussing the registration and mutual authentication phases, this subsection first presents the cryptographic notations used throughout the manuscript. The cryptographic notations are summarized in Table 1.

Table 1. Cryptography Notations

No	Notations	Descriptions
1	$User_{ID}$	User identity
2	PID_{US}	Pseudo identity
3	v_i	The nonce is updated in every session
4	$Secret_{K_{US}}$	Secret key between user and server
5	SK_{SU}	The session key is updated in every session
6	$PID_{US_{Syn}} = \{pid_{syn_1}, pid_{syn_2}, pid_{syn_3}, \dots pid_{syn_n}\}$ $Secret_{K_{US_{Syn}}} = \{Secret_{K_{syn_1}}, Secret_{K_{syn_2}}, Secret_{K_{S}}$	The pseudonym–secret key synchronization pair is used whenever desynchronization occurs
7	$(K_{\emptyset_U}, helpdata) = FE.Gen(\emptyset_u)$ $K_{\emptyset_U} = FE.Rec(\emptyset_u, helpdata)$	The fuzzy extractor (Generator and Reconstructor) is used to generate and reconstruct the biometric key from the fingerprint.

3.3.1. Registration Phase

1) Step 1: The user/smartcard sends the user identity and a registration request to the server: $User_{ID}, Registration_{Request}$.
2) Step 2: The server generates the following parameters: User secret key $Secret_{K_{US}}$, User pseudonym: PID_{US}, Synchronized pseudonym set: $\text{PID}_{US_{\text{Syn}}} = \{\text{pid}_{\text{syn}_1}, \text{pid}_{\text{syn}_2}, \text{pid}_{\text{syn}_3}, \dots \text{pid}_{\text{syn}_n}\}$ Synchronized secret keys: $Secret_{K_{US_{Syn}}} = \{Secret_{K_{\text{syn}_1}}, Secret_{K_{\text{syn}_2}}, Secret_{K_{\text{syn}_3}}, \dots Secret_{K_{\text{syn}_n}}\}$. The server stores: $User_{ID}$, $Secret_{K_{US}}$, PID_{US}, $\text{PID}_{US_{\text{Syn}}}$, $Secret_{K_{US_{Syn}}}$. Then the server sends: $\{User_{ID}, Secret_{K_{US}}, PID_{US}, \text{PID}_{US_{\text{Syn}}}, Secret_{K_{US_{Syn}}}\}$ to the user/smartcard over a secure channel.
3) Step 3: Upon receiving $\{Secret_{K_{US}}, PID_{US}, PID_{US_{Syn}}, Secret_{K_{US_{Syn}}}\}$, the user or smartcard: Imprints biometric: $\emptyset_u$. Generates fuzzy extractor output: $(K_{\emptyset_U}, helpdata) = FE.Gen(\emptyset_u)$. Protects the server-issued secrets using biometric key: $Secret_{K_{US}}^{enc} = Secret_{K_{US}} \oplus h(K_{\emptyset_U})$, $PID_{US}^{enc} = PID_{US} \oplus h(K_{\emptyset_U})$, $Secret_{K_{US_{Syn}}}^{enc} = Secret_{K_{US_{Syn}}} \oplus h(K_{\emptyset_U})$, $PID_{US_{Syn}}^{enc} = \text{PID}_{US_{\text{Syn}}} \oplus h(K_{\emptyset_U})$. The registration phase is securely completed.

3.3.2. Mutual Authentication Phase

Mutual authentication is performed between the vehicle (OBU/smart card) and the server when the vehicle requests services such as predictive vehicle maintenance, over-the-air (OTA) updates, fleet management, or safety enhancement. The authentication phase is illustrated in figure 1.

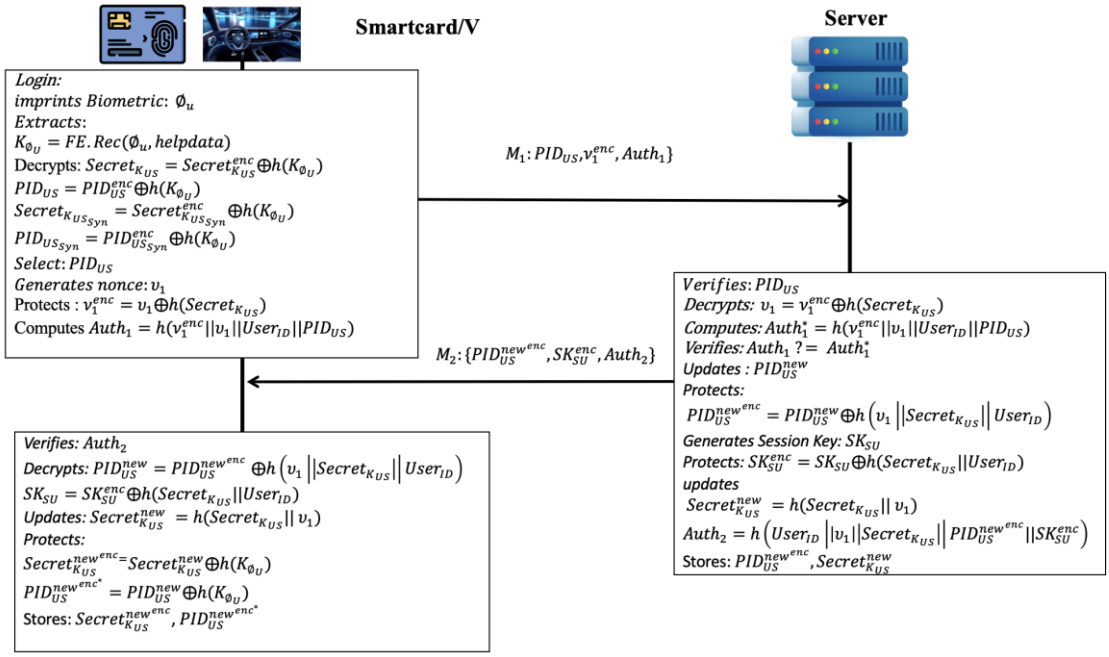

Figure 1. Authentication Phase

Step 1: User Login and Sends Message M₁ (User → Server). The user logs in and provides a biometric $\emptyset_u$. The smartcard reconstructs the fuzzy-extractor key: $K_{\emptyset_U} = FE.Rec(\emptyset_u, helpdata)$. Decrypt stored parameters: $Secret_{K_{US}} = Secret_{K_{US}}^{enc} \oplus h(K_{\emptyset_U})$, $PID_{US} = PID_{US}^{enc} \oplus h(K_{\emptyset_U})$, $Secret_{K_{US_{Syn}}} = Secret_{K_{US_{Syn}}}^{enc} \oplus h(K_{\emptyset_U})$, $\text{PID}_{US_{\text{Syn}}} = PID_{US_{Syn}}^{enc} \oplus h\left(K_{\emptyset_U}\right)$.The smartcard selects a pseudonym PID_{US} and generates a nonce v_1. Protects the nonce: $v_1^{enc} = v_1 \oplus h(Secret_{K_{US}})$. Computes: $Auth_1 = h(v_1^{enc}||v_1||User_{ID}||PID_{US})$The smartcard sends $M_1: PID_{US}, v_1^{enc}, Auth_1\}$to the server.

Step 2: Server Verification and Sends Message M₂ (Server → User). Upon receiving $M_1: PID_{US}, v_1^{enc}, Auth_1\}$, the server Verifies PID_{US}. Recovers the nonce: $v_1 = v_1^{enc} \oplus h\left(Secret_{K_{US}}\right)$. Computes: $Auth_1^* = h(v_1^{enc}||v_1||User_{ID}||PID_{US})$ and verifies $Auth_1 ? = Auth_1^*$. Updates the user pseudonym: PID_{US}^{new}. Protects the new pseudonym: $PID_{US}^{new^{enc}} = PID_{US}^{new} \oplus h\left(v_1 \left|\left|Secret_{K_{US}}\right|\right| User_{ID}\right)$. Generates the session key: SK_{SU}. Protects it: $SK_{SU}^{enc} = SK_{SU} \oplus h(Secret_{K_{US}}||User_{ID})$. Updates the user secret: $Secret_{K_{US}}^{new} = h(Secret_{K_{US}}|| v_1)$. Computes: $Auth_2 = h\left(User_{ID} \left||v_1||Secret_{K_{US}}\right|\right| PID_{US}^{new^{enc}}||SK_{SU}^{enc}\right)$. The server stores $PID_{US}^{new^{enc}}$, $Secret_{K_{US}}^{new}$ and sends: $M_2: \{PID_{US}^{new^{enc}}, SK_{SU}^{enc}, Auth_2\}$.

Step 3: User Final Verification and Update (User Side). Upon receiving $M_2: \{PID_{US}^{new^{enc}}, SK_{SU}^{enc}, Auth_2\}$, the user verifies $Auth_2$. Recovers the updated pseudonym: $PID_{US}^{new} = PID_{US}^{new^{enc}} \oplus h\left(v_1 \left|\left|Secret_{K_{US}}\right|\right| User_{ID}\right)$. Recovers the session key: $SK_{SU} = SK_{SU}^{enc} \oplus h(Secret_{K_{US}}||User_{ID})$. Updates the user secret: $Secret_{K_{US}}^{new} = h(Secret_{K_{US}}|| v_1)$. Protects stored

secrets for the next login: $Secret_{K_{US}}^{new^{enc}}$=$Secret_{K_{US}}^{new} \oplus h(K_{\phi_U})$, $PID_{US}^{new^{enc^*}} = PID_{US}^{new} \oplus h(K_{\phi_U})$. Stores: $Secret_{K_{US}}^{new^{enc}}$, $PID_{US}^{new^{enc^*}}$. The mutual authentication is successfully completed.

3.4. Fromal analysis

The formal analysis is conducted under the Real-or-Random (RoR) model (Zahednejad & Gao, 2023), (Lee et al., 2022), (Bagheri et al., 2024), (Saleem et al., 2024), (Ryu et al., 2022). This analysis presents the participants and their roles, the adversary's capabilities, the RoR game-based proof, and the semantic security of the session key.

3.4.1. Participants and their roles

The participants in the protocol are the user/smartcard and the server. A partnering relationship is established between these two entities when they authenticate each other and successfully preserve user privacy. Freshness is achieved by updating credentials in every session and using nonces, which enables the protocol to detect replay attacks and man-in-the-middle attacks even when an adversary attempts to compromise the communication.

3.4.2. Adversary Capabilities

$\prod_{execute}(\prod^{su}, \prod^{S})$: The attacker can eavesdrop on and intercept all messages exchanged between the user/smartcard and the server over the public channel.

$\prod_{send}(\prod^{su}, m)$: The attacker can conduct replay attacks by injecting or resending previously intercepted messages retrieved from the public channel.

$\prod_{corruptSmartcard}(\prod^{su})$: The attacker can steal the smartcard/device and perform side-channel analysis to extract sensitive information stored in the smartcard's memory.

$\prod_{test}(\prod^{su}, \prod^{S})$: The attacker attempts to guess the session key by issuing the Test query. The challenger returns either the real session key or a random value based on a probabilistic coin flip. If the attacker's guess satisfies C=C′, the attempt is considered successful; otherwise, it fails.

3.4.3. The proof the ROR model

Hypothesis: The hypothesis is constructed using the RoR game-based proof model, where the adversary $\hat{\mathcal{A}}$ is allowed to perform the following queries: $\prod_{execute}(\prod^{su}, \prod^{S})$, $\prod_{send}(\prod^{su}, m)$, $\prod_{corruptSmartcard}(\prod^{su})$, $\prod_{test}(\prod^{su}, \prod^{S})$. The advantage Adv of adversary $\hat{\mathcal{A}}$ in attacking the proposed protocol Pro within time T, where the protocol uses the hash function $\mathbb{H}$ and biometric key $\mathbb{B}$ with respective complexity bounds $\mathbb{h}^2$ and $\mathbb{b}^2$, and attempts to guess the session key (SK), is formally expressed as: $Adv_{\hat{\mathcal{A}}}^{Pro}(T) \leq \frac{\mathbb{h}^2}{|\mathbb{H}|} + \frac{\mathbb{b}^2}{|\mathbb{B}|} + 2Adv_{\hat{\mathcal{A}}}^{SK}(T)$, where $Adv_{\hat{\mathcal{A}}}^{SK}(T)$) must remain negligible (an infinitesimal value $\mathbb{u}$).

Proof: The proof is modeled using oracle-based game transitions. The analysis is divided into five games $Game_i$ (from G_0 to G_4). Each game represents the probability $\mathbb{P}$ of the adversary $\hat{\mathcal{A}}$ succeeding in guessing the correct credentials based on the Test query (coin flip). The general notation is: $\mathbb{P}\left[Succ_{\hat{\mathcal{A}}}^{Game_i}(T)\right]$.

Game G_0: Actual Attack Setting. This represents the real execution of the protocol in the presence of an adversary. The adversary attempts to guess the session key and relies on the coin flip C. Thus, the adversary's advantage is: $Adv_{\hat{A}}^{Pro}(T) = \left|2\mathbb{P}\left[Succ_{\hat{A}}^{Game_0}(T)\right] - 1\right|$.

Game G_1: Adversary Executes $\prod_{execute}(\prod^{su}, \prod^{S})$. Here, the adversary eavesdrops and intercepts messages exchanged between parties. However, since the messages are encrypted, the adversary gains no additional information. Hence: $\mathbb{P}\left[Succ_{\hat{A}}^{Game_1}(T)\right] = \mathbb{P}\left[Succ_{\hat{A}}^{Game_0}(T)\right]$.

Game G_2: Adversary Executes $\prod_{send}(\prod^{su}, \boldsymbol{m})$. The adversary performs replay attempts and tries to obtain credentials. Because: credentials are encrypted, credentials are updated every session, and one-way hash functions are collision resistant, the attacker cannot recover valid credentials. Therefore, the probability difference is bounded by: $\left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_2}(T)\right] - \mathbb{P}\left[Succ_{\hat{A}}^{Game_1}(T)\right]\right| \leq \frac{\mathbb{h}^2}{2|\mathbb{H}|}$.

Game G_3: Adversary Executes $\prod_{corruptSmartcard}(\prod^{su})$. The adversary steals the smartcard and attempts a side-channel attack. However, since: the biometric is unique and the biometric-derived key cannot be generated by the adversary, the advantage difference is bounded by: $\left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_3}(T)\right] - \mathbb{P}\left[Succ_{\hat{A}}^{Game_2}(T)\right]\right| \leq \frac{\mathbb{b}^2}{2|\mathbb{B}|}$.

Game G_4: Adversary Executes $\prod_{test}(\prod^{su}, \prod^{S})$. The adversary attempts to distinguish the real session key from a random key by flipping a coin. Since: the protocol preserves session-key secrecy, adversaries cannot infer the key from public messages, and credentials remain encrypted even after smartcard theft, we have: $\left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_4}(T)\right] - \mathbb{P}\left[Succ_{\hat{A}}^{Game_3}(T)\right]\right| \leq Adv_{\hat{A}}^{SK}(T)$. Because the Test query returns a real or random session key with equal probability: $\mathbb{P}\left[Succ_{\hat{A}}^{Game_4}(T)\right] = \frac{1}{2}$.

Combining the Results of Games G_0–G_4. $\frac{1}{2}Adv_{\hat{A}}^{Pro}(T) = \left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_4}(T)\right] = \frac{1}{2}\right| = \left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_0}(T)\right] - \mathbb{P}\left[Succ_{\hat{A}}^{Game_4}(T)\right]\right| \leq \sum_{i=0}^{3}\left|\mathbb{P}\left[Succ_{\hat{A}}^{Game_{i+1}}(T)\right] - \mathbb{P}\left[Succ_{\hat{A}}^{Game_i}(T)\right]\right| = \frac{\mathbb{h}^2}{2|\mathbb{H}|} + \frac{\mathbb{b}^2}{2|\mathbb{B}|} + Adv_{\hat{A}}^{SK}(T)$. Therefore, the complete notation of the adversary's probability of revealing the session key in the proposed protocol is expressed as follows, which confirms the formulated hypothesis: $Adv_{\hat{A}}^{Pro}(T) \leq \frac{\mathbb{h}^2}{|\mathbb{H}|} + \frac{\mathbb{b}^2}{|\mathbb{B}|} + 2Adv_{\hat{A}}^{SK}(T)$.

3.4.4. Semantic security

Theorem 1: The proposed authentication protocol provides mutual authentication. Proof: Both parties in the protocol are able to authenticate each other. In Step 1, the user/smartcard sends M_1: $\{PID_{US}, v_1^{enc}, Auth_1\}$, and the server verifies $Auth_1$ by checking PID_{US}, $User_{ID}$, and v_1. The server then sends M_2: $\{PID_{US}^{new^{enc}}, SK_{SU}^{enc}, Auth_2\}$, which is validated by the user/smartcard. Since each party successfully verifies the authentication message received from the other, the proposed protocol achieves secure mutual authentication.

Theorem 2: The authentication protocol achieves strong anonymity, meaning that it simultaneously provides anonymity, unlinkability, and untraceability. When a protocol satisfies all three properties at the same time, it is said to achieve strong anonymity. Proof: The protocol never transmits the actual user identity over the public channel; therefore, even if an attacker eavesdrops on or intercepts the communication, the real identity cannot be obtained. Furthermore, all secret credentials are updated in every session, and the pseudonymous identity is refreshed each time. Even if an attacker collects public-channel data or performs a stronger attack, such as stealing the device, the stored credentials remain protected by the biometric-derived key and cannot be extracted. Consequently, the protocol simultaneously ensures unlinkability, untraceability, and anonymity, thereby achieving strong anonymity.

Theorem 3: The protocol achieves session key security even if an attacker steals the device or conducts a side-channel attack. Proof: The session key is generated by the server and is protected using a secret key. Both the session key and the secret key are updated in every session, ensuring freshness and resistance to compromise. During communication, the server sends SK_{SU}^{enc} to the user; even if an attacker intercepts this value from the public channel, they cannot derive the session key because they do not possess the corresponding secret key required for decryption. Likewise, stealing the device or performing a side-channel attack does not help the attacker, as all stored credentials are protected using a biometric-derived key. Therefore, since the attacker has no feasible means to obtain or decrypt the session key, the protocol successfully achieves session key security.

Theorem 4: The proposed protocol achieves perfect forward and backward secrecy, even if an attacker steals the smartcard and obtains the data stored in its memory. Proof: If an attacker performs a strong attack by stealing the smartcard and extracting all data from its memory, the attacker can only obtain encrypted information, since every stored credential is protected by a biometric-derived key. Moreover, the protocol updates all critical credentials—including pseudonyms, secret keys, and session keys—in every session. As a result, the attacker has no opportunity to recover past session keys (forward secrecy) or future session keys (backward secrecy), because the previous and subsequent secrets cannot be derived from the compromised data. Therefore, the protocol successfully achieves perfect forward and backward secrecy.

Theorem 5: The proposed protocol withstands replay attacks. Proof: If an attacker intercepts the message M_1: PID_{US},v_1^{enc}, $Auth_1$} from the public channel and replays it in a later session, the protocol can easily detect that the message is outdated. This is because the protocol updates all critical values in every session: the user generates a new nonce v_1^{new}, a new pseudonymous identity PID_{US}^{new}, and a new secret key $Secret_{K_{US}}^{new}$. The new encrypted nonce is computed as $v_1^{new^{enc}} = v_1^{new} \oplus h(Secret_{US}^{new})$. Upon receiving a replayed message, the server verifies the authentication tag by comparing $Auth_1 = h(v_1^{enc}||v_1||User_{ID}||PID_{US})$ with the newly expected value $Auth_{new_1} = h(v_1^{new^{enc}}||v_1^{new}||User_{ID}||PID_{US}^{new})$. Since $Auth_1 = !\,Auth_{new_1}$, the server immediately detects that the message is a replay of a previous session. Therefore, the proposed protocol effectively withstands replay attacks.

Theorem 6: The proposed protocol withstands man-in-the-middle (MitM) attacks. Proof: Similar to the replay attack scenario, a MitM attacker may intercept $Auth_1 = h(v_1^{enc}||v_1||User_{ID}||PID_{US})$ from the public channel and then attempt to impersonate the server. The fake server may generate forged values such

as $PID_{US}^{new^{fake}}$, $Auth_{fake_2} = h\left(PID_{US-fake}^{new^{enc}} || SK_{SU-fake}^{enc}, \upsilon_{fake}\right)$, and then send $\{PID_{US}^{new^{fake}}, Auth_{fake_2}\}$ to the user. However, the legitimate user can easily detect the attack because $PID_{US}^{new^{fake}} != PID_{US}^{new}$ and $Auth_{fake_2} ! = Auth_2$ Since the attacker cannot derive the correct pseudonym, secret key, or authentication tag, the forged message fails verification. Therefore, the proposed protocol successfully withstands man-in-the-middle attacks.

Theorem 7: The proposed protocol withstands physical attacks. Proof: If an attacker steals the device and attempts to extract the actual credentials stored in memory, the attack will fail because all stored credentials are protected using the biometric-derived key $K_{\emptyset_U}$. Since the attacker has no means to obtain or reconstruct the user's biometric key, they cannot decrypt or recover any sensitive information stored on the device. Therefore, the proposed protocol effectively withstands physical attacks.

Theorem 8: The proposed protocol withstands impersonation attacks. Proof: Even if an attacker intercepts messages from the public channel and steals the smartcard, they still cannot obtain the actual identity of the user because the protocol employs pseudonymous identities that are updated in every session. This makes it extremely difficult for the attacker to derive or reconstruct the real user identity. Since impersonation requires knowledge of the legitimate identity and valid credentials—both of which remain protected and inaccessible—the proposed protocol successfully withstands impersonation attacks.

Theorem 9: The proposed protocol withstands password-guessing attacks. Proof: The proposed protocol does not use a password as an authentication factor; instead, it relies on the uniqueness of the user's fingerprint or biometric features for authentication. Since no password is involved in the authentication process, an attacker has no opportunity to perform password-guessing or dictionary-based attacks. Therefore, the protocol effectively prevents and withstands password-guessing attacks.

Theorem 10: The proposed protocol resolves desynchronization problems. Proof: If desynchronization occurs, the user/smartcard initiates communication with the server by sending the synchronized pseudonym pid_{syn_1} along with a nonce encrypted using the synchronized secret key. Specifically, the smartcard generates a synchronization nonce υ_{syn_1} and protects it as $\upsilon_{syn}^{enc} = \upsilon_{syn_1} \oplus h(Secret_{K_{syn_1}})$. It then computes the synchronization authentication code $Auth_{syn} = h(pid_{syn_1} || \upsilon_{syn_1} || \upsilon_{syn}^{enc})$ and sends the message M_{Syn}: $\{pid_{syn_1}, \upsilon_{syn}^{enc}, Auth_{syn}\}$ to the server. The server verifies the authentication code; if $Auth_{syn}^{*} = Auth_{syn}$, the synchronization succeeds, otherwise the process fails and communication is aborted. Therefore, by using synchronized pseudonyms, encrypted synchronization nonces, and authentication codes, the protocol is capable of resolving desynchronization issues while preserving strong anonymity.

Theorem 11: The proposed protocol withstands denial-of-service (DoS) attacks. Proof: Because the protocol is capable of resolving desynchronization issues, even if an attacker attempts to flood the network and temporarily disrupt communication, either party can re-establish synchronization by sending the synchronization variables. This ensures that the service remains available and communication can continue. Since the protocol can automatically recover from desynchronization, as demonstrated in Theorem 10, it inherently provides resistance to DoS attacks. Therefore, the proposed protocol successfully withstands denial-of-service attacks.

3.5. Formal Security Verification Using Scyther

To further strengthen the security analysis of the proposed authentication protocol, formal security verification was performed using the Scyther tool. Scyther is an automated protocol verification tool that analyzes authentication protocols under the Dolev–Yao adversary model (Cremers, 2014), (Gerenli et al., 2026), (Saleem et al., 2024), (Shariq et al., 2024), (Ghosh et al., 2025). The verification results of the proposed authentication protocol are presented in figure 2.

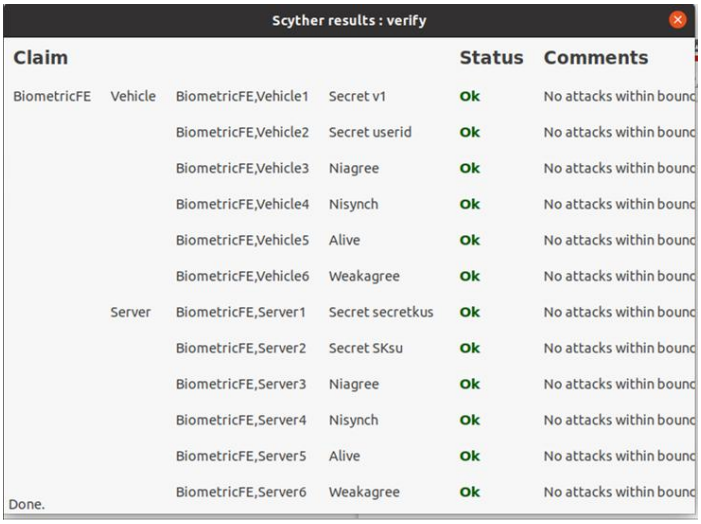

Figure 2. Scyther Verification Results

Figure 2 presents the formal verification results obtained using the Scyther tool. The verification results indicate that all specified security claims are successfully satisfied, and no attacks were detected within the verification bounds. Under the Dolev–Yao adversary model, the Scyther tool verifies the protocol against network-based attacks, including impersonation, replay, and man-in-the-middle attacks, while also confirming the authentication properties represented by Alive, Weakagree, Niagree, and Nisynch. Furthermore, the secrecy claims for the user identity, session key, and secret values are successfully verified. These results provide additional evidence that the proposed authentication protocol is secure against the considered adversarial capabilities.

3.6. Comparison of Security Features and Execution Time

This section compares the proposed authentication protocol with existing authentication protocols in terms of security features and execution time. The security features (SF) considered in this comparison include SF1: mutual authentication, SF2: strong anonymity, SF3: session key security, SF4: perfect forward and backward secrecy, SF5: resistance to replay attacks, SF6: resistance to denial-of-service (DoS) attacks, SF7: resistance to physical attacks, and SF8: resistance to password guessing attacks.

For the execution time evaluation, the cryptographic primitives were implemented and evaluated on an Arduino Uno R3 using the Tinkercad simulation environment. The evaluated cryptographic primitives include SHA-256, Physical Unclonable Function (PUF), Elliptic Curve Cryptography (ECC), Rivest–Shamir–Adleman (RSA), and the Fuzzy Extractor (FE). The measured execution times of the corresponding cryptographic operations are T_{SHA256}: 17.54 ms, T_{PUF}: 3.16 ms, T_{ECC}: 38.13 ms, T_{RSA}: 329.29 ms, T_{FE}: 17.13 ms, respectively.

Table 2. Comparison of Security Features

The protocol	SF1	SF2	SF3	SF4	SF5	SF6	SF7	SF8
(Sibahee et al., 2024)	√	√	√	X	√	X	X	X
(J. Wang et al., 2022)	√	X	√	X	√	√	X	√
(Lin & Yang, 2026)	√	X	√	√	√	√	X	√
(Huang & Chen, 2026)	√	X	√	X	√	√	X	X
Porposed Protocol	√	√	√	√	√	√	√	√

Table 2 compares the security features of the proposed authentication protocol with representative authentication protocols from the literature (Sibahee et al., 2024), (J. Wang et al., 2022), (Lin & Yang, 2026), (Huang & Chen, 2026). As shown in Table 2, the proposed authentication protocol satisfies all eight security features. In contrast, the compared authentication protocols satisfy only a subset of the considered security features. Specifically, (Sibahee et al., 2024) and (Huang & Chen, 2026) satisfy four security features, (J. Wang et al., 2022) satisfies five security features, and (Lin & Yang, 2026) satisfies six security features. Therefore, the proposed authentication protocol provides more comprehensive security protection than the compared authentication protocols.

Table 3. Comparison of Execution Time

Protocol	Vehicle/Smart Card (ms)	Server (ms)	Total (ms)
(Sibahee et al., 2024)	$12T_{SHA256}$=210.48	$18T_{SHA256}$=315.72	526.20
(J. Wang et al., 2022)	$6T_{ECC} + 6T_{SHA256}$=334.02	$8T_{ECC} + 5T_{SHA256} = 392.74$	726.76
(Lin & Yang, 2026)	$T_{PUF} + 4T_{RSA} + T_{ECC}$=1,358.45	$T_{PUF}+2T_{RSA} + T_{ECC}$=699.87	2,058.32
(Huang & Chen, 2026)	$2T_{ECC} + T_{RSA} + 7T_{SHA256}$=528.33	$2T_{ECC} + T_{RSA} + +7T_{SHA256}$=528.33	1,056.66
Proposed protocol	$7T_{SHA256} + T_{FE}$=139.91	$4T_{SHA256}$=210.07	210.07

Table 3 compares the execution time of the proposed authentication protocol with representative authentication protocols from the literature. The execution time of each cryptographic primitive was measured on the same experimental platform, and the total execution time of each protocol was calculated based on the number of cryptographic operations required. The proposed authentication protocol requires 139.91 ms on the vehicle/smart card side and 70.16 ms on the server side, resulting in a total execution time of 210.07 ms, which is lower than those of Sibahee et al. (526.20 ms), Wang et al. (726.76 ms), Huang and Chen (1,056.66 ms), and Lin and Yang (2,058.32 ms). The lower execution time is mainly attributed to the reduced computational overhead of the proposed authentication protocol, which requires fewer cryptographic operations than the compared authentication protocols while preserving the required security properties. Consequently, the proposed authentication protocol is more suitable for resource-constrained vehicular environments.

4. CONCLUSIONS AND RECOMMENDATIONS

This manuscript presents a cryptanalysis of the authentication protocol proposed by Sibahee et al., identifies its security weaknesses, and proposes a new authentication protocol to address the identified limitations. Formal security analysis based on the Real-or-Random (RoR) model demonstrates that the proposed authentication protocol achieves strong security properties, including mutual authentication, strong anonymity, perfect forward and backward secrecy, and resilience against desynchronization. In addition, the proposed authentication protocol provides resistance against replay, impersonation, denial-of-

service (DoS), man-in-the-middle, and physical attacks. Formal verification using the Scyther tool further confirms that all specified security claims are successfully satisfied without detecting any attacks. Furthermore, the execution time evaluation shows that the proposed authentication protocol achieves the lowest execution time among the compared authentication protocols, demonstrating its suitability for resource-constrained IoV devices. Future work will focus on implementing and deploying the proposed authentication protocol in real-world IoV environments to validate its practicality, scalability, and performance under realistic communication conditions.

ACKNOWLEDGEMENT

The author gratefully acknowledges the support of the Faculty of Computer Science, Universitas Singaperbangsa Karawang. The author also sincerely thanks the editor and the anonymous reviewers for their insightful comments and constructive suggestions that helped improve this manuscript.

REFERENCES

Aman, M. N., & Javaid, U. (2021). A Privacy-Preserving and Scalable Authentication Protocol for the Internet of Vehicles. *IEEE Internet of Things Journal*, *8*(2), 1123–1139.

Bagheri, N., Bendavid, Y., Safkhani, M., & Rostampour, S. (2024). Smart Grid Security: A PUF-Based Authentication and Key Agreement Protocol. *Future Internet*, *16*(1), 1–18. https://doi.org/10.3390/fi16010009

Cremers, C. (2014). Scyther User Manual. *The CISPA Helmholtz Center for Information Security*, 2–52.

Dodis, Y., Reyzin, L., & Smith, A. (2004). Fuzzy Extractors : How to Generate Strong Keys from Biometrics and Other Noisy Data. *In Advances in Cryptology— EUROCRYPT'2004 (Lecture Notes in Computer Science). Heidelberg, German: Springer*, 523–540.

Dolev, D., & Yao, A. (1983). On the Security of Public Key Protocols. *IEEE TRANSACTIONS ON INFORMATION THEORY*, *M*, 198–208.

Gerenli, O., Karabulut-kurt, G., & Ozdemir, E. (2026). A user centric group authentication scheme for secure communication. *Scientific Reports*, 1–19.

Ghosh, H., Maurya, P. K., Bagchi, S., & Dwivedi, A. D. (2025). Lightweight RFID-enabled authentication protocol in post-quantum environment. *Computers and Electrical Engineering*, *124*(PB), 110367. https://doi.org/10.1016/j.compeleceng.2025.110367

Huang, W., & Chen, S. (2026). Baa-iov : Blockchain-enabled anonymous authentication for internet of vehicles. *PLoS ONE*, 1–34. https://doi.org/10.1371/journal.pone.0347787

Javaid, U., & Aman, M. N. (2020). A Scalable Protocol for Driving Trust Management in Internet of Vehicles With Blockchain. *IEEE Internet of Things Journal*, *7*(12), 11815–11829.

Khan, B., Malip, A., & Khan, A. (2026). Post-quantum blind signature-based authentication for intelligent transportation systems. *Computer Networks*, *287*(January), 112513. https://doi.org/10.1016/j.comnet.2026.112513

Kumar, N., & Chaudhary, A. (2026). Secure and Lightweight Mechanism for UAV-GCS and UAV-UAV Authentication Based on PUF. *IEEE Internet of Things Journal*, *PP*(April 2025), 1. https://doi.org/10.1109/JIOT.2026.3669029

Kumar, S., Shariq, M., & Singh, G. (2026). An ultralightweight and reliable authentication protocol for secure communication in UAV-assisted IoAV systems. *Computers and Electrical Engineering*, *132*(October 2025), 110998. https://doi.org/10.1016/j.compeleceng.2026.110998

Lee, J. Y., Oh, J. H., Kwon, D. K., Kim, M. H., Yu, S. J., Jho, N. S., & Park, Y. (2022). PUFTAP-IoT: PUF-Based Three-Factor Authentication Protocol in IoT Environment Focused on Sensing Devices. *Sensors*, *22*(18), 1–24. https://doi.org/10.3390/s22187075

Lin, H., & Yang, H. (2026). Blockchain-PUF based two-factor authentication and key agreement scheme in IoV. *Journal of Information Security and Applications*, *100*(May), 104494. https://doi.org/10.1016/j.jisa.2026.104494

Ryu, J., Oh, J., Kwon, D., Son, S., Lee, J., Park, Y., & Park, Y. (2022). Secure ECC-Based Three-Factor Mutual Authentication Protocol for Telecare Medical Information System. *IEEE Access*, *10*, 11511–11526. https://doi.org/10.1109/ACCESS.2022.3145959

Saleem, M. A., Li, X., Mahmood, K., Tariq, T., Alenazi, M. J. F., & Das, A. K. (2024). Secure RFID-Assisted Authentication Protocol for Vehicular Cloud Computing Environment. *IEEE Transactions on Intelligent Transportation Systems*, *25*(9), 12528–12537. https://doi.org/10.1109/TITS.2024.3371464

Shariq, M., Conti, M., Singh, K., Lal, C., Das, A. K., Chaudhry, S. A., & Masud, M. (2024). Anonymous and reliable ultralightweight RFID-enabled authentication scheme for IoT systems in cloud computing. *Computer Networks*, *252*(July), 110678. https://doi.org/10.1016/j.comnet.2024.110678

Shi, D., Nie, X., Xu, M., Cheng, H., & Alam, M. (2025). A secure and efficient lattice-based conditional privacy-preserving authentication protocol for the VANET. *Vehicular Communications*, *55*(November 2024), 100958. https://doi.org/10.1016/j.vehcom.2025.100958

Sibahee, M. A. Al, Nyangaresi, V. O., & Abduljabbar, Z. A. (2024). Two-Factor Privacy-Preserving Protocol for Efficient Authentication in Internet of Vehicles Networks. *IEEE Internet of Things Journal*, *11*(8), 14253–14266. https://doi.org/10.1109/JIOT.2023.3340259

Singh, G., Sharma, S., Khader, A., Saudagar, J., & Kumar, S. (2026). A secure group-based authentication protocol for IoVT in 5G-enabled smart transportation and road safety systems. *Scientific Reports*, 1–24.

Tiwari, V. K., & Kumar, P. (2026). Secure authentication protocol for internet of vehicles using blockchain based authorized user detection. *Peer-to-Peer Networking and Applications*, *5*.

Wang, J., Wu, L., Wang, H., Choo, K. K. R., Wang, L., & He, D. (2022). A Secure and Efficient Multiserver Authentication and Key Agreement Protocol for Internet of Vehicles. *IEEE Internet of Things Journal*, *9*(23), 24398–24416. https://doi.org/10.1109/JIOT.2022.3188731

Wang, X., & Xie, Y. (2025). An improved biometric authentication and key agreement scheme based on fuzzy extractor for Wireless Body Area Networks. *Journal of Information Security and Applications*, *91*(April), 104047. https://doi.org/10.1016/j.jisa.2025.104047

Wen, Y., & Su, Y. (2025). Post-Quantum Secure Multi-Factor Authentication Protocol. *Entropy*, 1–20. https://doi.org/https://doi.org/10.3390/e27070765

Xue, K., Luo, X., Ma, Y., Li, J., Liu, J., & Wei, D. S. L. (2022). A Distributed Authentication Scheme Based on Smart Contract for Roaming Service in Mobile Vehicular Networks. *IEEE Transactions on Vehicular Technology*, *71*(5), 5284–5297. https://doi.org/10.1109/TVT.2022.3148303

Yafoz, A., Alsini, R., & Almagrabi, A. O. (2026). LEAP : VANET : A lightweight and efficient authentication protocol for intelligent transportation system using VANET. *ICT Express*, *February*, 0–5. https://doi.org/10.1016/j.icte.2026.03.013

Zahednejad, B., & Gao, C. zhi. (2023). A secure and efficient AKE scheme for IoT devices using PUF and cancellable biometrics. *Internet of Things (Netherlands)*, *24*(April), 100937. https://doi.org/10.1016/j.iot.2023.100937

Zhang, H., Lai, Y., & Chen, Y. (2023). Authentication methods for internet of vehicles based on trusted connection architecture. *Simulation Modelling Practice and Theory*, *122*(June 2022), 102681. https://doi.org/10.1016/j.simpat.2022.102681