mathematics

MDPI

*Article*

# Blockchain-Based Secure Authentication Protocol for Fog-Enabled IoT Environments

**Taehun Kim [1], Deokkyu Kwon [1], Yohan Park [2,*] and Youngho Park [1,*]**

[1] School of Electronic and Electrical Engineering, Kyungpook National University, Daegu 41566, Republic of Korea; kimth028@knu.ac.kr (T.K.); kdk145@knu.ac.kr (D.K.)
[2] School of Computer Engineering, Keimyung University, Daegu 42601, Republic of Korea
* Correspondence: yhpark@kmu.ac.kr (Y.P.); parkyh@knu.ac.kr (Y.P.); Tel.: +82-53-950-7842 (Youngho Park)

**Abstract**

Fog computing technology grants computing and storage resources to nearby IoT devices, enabling a fast response and ensuring data locality. Thus, fog-enabled IoT environments provide real-time and convenient services to users in healthcare, agriculture, and road traffic monitoring. However, messages are exchanged on public channels, which can be targeted to various security attacks. Hence, secure authentication protocols are critical for reliable fog-enabled IoT services. In 2024, Harbi et al. proposed a remote user authentication protocol for fog-enabled IoT environments. They claimed that their protocol can resist various security attacks and ensure session key secrecy. Unfortunately, we have identified several vulnerabilities in their protocol, including to insider, denial of service (DoS), and stolen verifier attacks. We also prove that their protocol does not ensure user untraceability and that it has an authentication problem. To address the security problems of their protocol, we propose a security-enhanced blockchain-based secure authentication protocol for fog-enabled IoT environments. We demonstrate the security robustness of the proposed protocol via informal and formal analyses, including Burrows–Abadi–Needham (BAN) logic, the Real-or-Random (RoR) model, and Automated Verification of Internet Security Protocols and Applications (AVISPA) simulation. Moreover, we compare the proposed protocol with related protocols to demonstrate the excellence of the proposed protocol in terms of efficiency and security. Finally, we conduct simulations using NS-3 to verify its real-world applicability.

**Keywords:** Internet of Things; fog computing; authentication; cryptanalysis; blockchain

**MSC:** 68M12



Academic Editors: Jianhua He, Yipeng Zhou, Wei Wang, Fan Wu, Antanas Cenys, Raymond Lee and Marjan Mernik

Received: 7 May 2025
Revised: 12 June 2025
Accepted: 29 June 2025
Published: 30 June 2025

**Citation:** Kim, T.; Kwon, D.; Park, Y.; Park, Y. Blockchain-Based Secure Authentication Protocol for Fog-Enabled IoT Environments. *Mathematics* **2025**, *13*, 2142. https://doi.org/10.3390/math13132142



## 1. Introduction

With the rapid development of Internet of Things (IoT) technology, use of IoT devices is expanding to diverse fields such as smart agriculture [1], the Industrial IoT [2], and healthcare services [3]. In addition, IoT devices are arranged to collect information from their surroundings. Large amounts of data are generated as the number of IoT devices increases. Thus, cloud computing can be employed because it can provide ample storage and high computing resources for the generated data [4]. Cloud computing can also offer interoperability for heterogeneous IoT devices [5]. However, cloud servers face several challenges in real-time IoT applications. Cloud servers are typically located in remote areas far from IoT devices, causing high latency [6]. The centralized architecture of the

cloud server can cause problems, including network congestion and single points of failure [7]. Considering these limitations, studies have been searching for better solutions for IoT environments.

Fog computing is a technology that extends cloud computing to the edge of the network [8]. In fog computing, fog nodes provide sufficient computing power, storage space, and networking services to nearby IoT devices, helping to share the loads of the cloud server. Fog computing is suitable for IoT environments because the local data collected by IoT devices can be processed and stored in fog nodes instead of transmitting them to cloud servers [9]. The network participation of fog nodes allows services to be distributed among nodes, resulting in low latency and real-time availability [10]. For example, drivers can access real-time data stored in fog nodes to choose the quickest path using a road traffic monitoring service [11]. In addition, doctors can access fog nodes to read real-time sensor data for diagnoses [12]. With fog computing technology, IoT systems provide instantaneous and localized services to users.

Nevertheless, security problems in fog-enabled IoT environments remain. Fog-enabled IoT environments generally comprise three entities: users, fog nodes, and a cloud server. Communication between users, nodes, and the cloud server is realized through a public channel; therefore, malicious attackers can conduct various security attacks, potentially putting the system at risk. For instance, attackers can collect and employ exchanged messages to impersonate a legitimate user and extract secret information. Attackers can also capture transmitted messages to reveal the user's identity, compromising user untraceability. In addition, attackers can steal a legitimate user's smart device and obtain stored parameters via power analysis [13]. Moreover, attackers can send an excessive number of requests to take down the system, preventing users from accessing IoT services [14]. However, blockchain can be used to improve security for fog-enabled IoT environments [15]. Blockchain is a distributed ledger managed by a peer-to-peer network which features privacy, decentralization, and immutability. Blockchain technology can be applied to fog-enabled IoT environments, as fog nodes are distributed on the network. As a peer node in the blockchain network, fog nodes share encrypted private user information for user validation during the authentication process [16]. By implementing blockchain, various security challenges that arise from centralized architecture of cloud computing can be mitigated. Consequently, to protect data privacy and ensure the reliability of services, a secure authentication protocol is critical for fog-enabled IoT environments.

In 2024, Harbi et al. [17] proposed a remote user authentication protocol for fog computing. Harbi et al. claimed that their protocol offers an efficient user authentication process and is secure against many security attacks, including impersonation, offline guessing, privileged insider, and man-in-the-middle (MitM) attacks. Moreover, they argued that their protocol can verify users using blockchain. However, we found that Harbi et al.'s protocol is not secure against insider, stolen verifier, or DoS attacks. Moreover, we discovered that their protocol cannot ensure untraceability of users and that it fails to validate the fog node, leading to unsuccessful authentication and session key agreement. To address these security vulnerabilities, we propose a security-enhanced authentication protocol using blockchain for fog-enabled IoT environments. The proposed protocol applies a fuzzy extractor to use biometric information, adding protection against offline guessing attacks. We also use lightweight cryptographic operations to reduce the load on our system, including exclusive-OR operations and hash functions. The main contributions of this paper are summarized as follows:

- We review the protocol by Harbi et al. and demonstrate its vulnerabilities to insider, DoS, and stolen verifier attacks. Moreover, we reveal that their protocol does not ensure untraceability and has an authentication problem.

- We propose a blockchain-based secure authentication protocol for IoT environments that resolves the security problems in the protocol. We use exclusive-OR operations and hash functions to retain low computational cost. In addition, we use a fuzzy extractor to implement users' biometric information for resistance against offline guessing attacks [18].
- We employ the Burrows–Abadi–Needham (BAN) logic [19], the Real-or-Random (ROR) model [20], and the Automated Verification of Internet Security Protocols and Applications (AVISPA) simulation tool [21,22] to demonstrate that the proposed protocol ensures mutual authentication and security of the session key and resists replay and MitM attacks. We also prove that the proposed protocol is secure against known security attacks, including desynchronization, session-specific random number leakage, insider attacks, and impersonation attacks, and that it ensures mutual authentication and user anonymity via informal analysis.
- We evaluate the security properties and computation and communication costs of the proposed protocol with relevant protocols. Moreover, we simulate the proposed protocol using Network Simulator 3 (NS-3) [23] to verify its capability in real-world environments.

*Article Organization*

Section 2 introduces the research background of authentication protocols in fog-enabled IoT environments. Section 3 describes core technologies addressed in this paper along with the adversary model and system model. Section 4 provides a summary of the protocol by Harbi et al., while Section 5 confirms the security vulnerabilities of their protocol. Section 6 presents the protocol proposed in this paper to overcome the security vulnerabilities of the previous protocol. Section 7 demonstrates the security of our proposed protocol using formal and informal analyses, while Section 8 presents performance analyses and NS-3 simulation results. Finally, Section 9 concludes and summarizes this work.

## 2. Research Background and Related Works

### *2.1. Research Background*

In 2012, Cisco [24] introduced the concept of fog computing, which is an extension of cloud computing that enables computing closer to IoT and end-user devices. Fog computing provides virtualized computation and storage resources between end devices and cloud servers; hence, it is considered a good option for IoT applications [6]. Subsequently, much research on integration of fog computing with IoT has been conducted. Peter [10] showed that fog computing has sufficient resources to handle the data produced by IoT devices and is able to provide real-time services. Hazra et al. [25] summarized several past studies on fog computing, presented a use case scenario in IoT–fog–cloud environments, and highlighted the requirement of security and privacy for fog computing. Burhan et al. [26] explained that fog-based IoT networks lead to security challenges around secure data sharing and protection of sensitive data. In below, we list general security requirements which are crucial to fog-enabled IoT services:

- *Authentication*: Users, fog nodes, and the cloud server should be mutually authenticated before allowing access to databases or sharing important information.
- *Availability*: Fog-enabled IoT services should be available to every legitimate user even under DoS attacks.
- *Confidentiality*: Adversaries or non-registered users should not be able to extract any useful information from messages.
- *Freshness*: Every transmitted message should be unique, i.e., not be identical with old messages.

- *Integrity*: Messages should not be modified during transmission.

### 2.2. Related Works

In 2018, Jia et al. [27] designed an authenticated key agreement protocol for IoT healthcare systems. In their protocol, fog nodes filter raw data sent by IoT healthcare devices and transmit the collected data to the cloud server for more operations, thereby enabling real-time response for healthcare services. In 2019, Ma et al. [28] suggested an authentication protocol for VANET using Elliptic Curve Cryptography (ECC). Fog nodes connect with vehicle users and the cloud to provide authentication messages and establish a session key. However, Eftekhari et al. [29] discovered that their protocol is vulnerable to insider, stolen smart card, and "honest-but-curious" fog server attacks. To address these security vulnerabilities, Eftekhari et al. proposed a security-enhanced protocol using fog servers. In 2021, Smet et al. [30] designed an authentication protocol using Physical Unclonable Functions (PUFs). In their protocol, the fog nodes forward users' requests to the cloud server and verify the users. In 2023, Saleem et al. [31] proposed a physically secure Authentication and Key Agreement (AKA) protocol using fuzzy extractors. In their scheme, Road-Side Units (RSU) communicate with a Trusted Party Agent (TPA) over a wired channel. The location area identifier $LID_r$ of the RSU is utilized for secure authentication between the entities. Tahir et al. [32] proposed a multi-factor authentication protocol using biometric information for VANET. Their protocol uses RSUs to relay authentication messages between vehicles and the TPA. Qiao et al. [33] designed an AKA protocol for HIoT using the Chebyshev polynomial. Users have healthcare IoT devices connected to fog nodes, which offer real-time responses for time-sensitive tasks. Some data are sent to the cloud server for additional analyses. In 2024, Irshad et al. [34] proposed an authentication protocol for HIoT. In their protocol, fog nodes assist in the processing of patients' diagnostic medical reports in real time.

Various authentication protocols for fog-enabled IoT environments using blockchain have been proposed to establish secure communication channels. In 2021, Tomar et al. [35] suggested an AKA protocol for the smart grid. In this protocol, fog nodes act as nodes in a blockchain and provide services using data from smart meters. Fog nodes are also responsible for providing authentication messages between smart meters and the cloud server. In 2023, Subramani et al. [36] proposed an anonymous authentication protocol for VANETs. Vehicles and fog nodes register with the Trusted Authority (TA), and the vehicle and fog node authenticate each other in the authentication phase. They claimed that this protocol can prevent physical attacks and ensure anonymity of users. Moreover, Zhang et al. [11] designed a privacy-preserving traffic route management protocol for VANETs. Their protocol utilized fog nodes for vehicle verification, aggregation of routes of vehicles, and real-time traffic warnings. However, Zhang et al.'s protocol can have a centralization problem because the TA supervises the entire network [37]. Wei et al. [38] suggested an AKA protocol with a multi-TA model for VANETs. They asserted that some intelligent transport system applications are enabled by fog nodes, such as intelligent traffic lights and real-time traffic warnings. In 2024, Tomar et al. [39] designed authentication protocol for VANET using the Chebyshev polynomial. In their protocol, fog servers provide data processing, verify vehicles, and participate in the key agreement process for secure data transfer. Subramani et al. [40] suggested an anonymous authentication protocol for the smart grid. In their protocol, fog nodes communicate with smart meters to verify the accuracy of electricity usage data and send the data along with a signature to the cloud blockchain server. Alsaeed et al. [41] proposed a group authentication protocol for IoMT. Their protocol utilizes a global blockchain for decentralized identification and registration. In addition, local blockchains managed by fog servers execute group authentication of

medical devices. Fog nodes in the same area provide computing and storage resources to a group of medical devices.

In 2024, Harbi et al. [17] proposed a remote user authentication protocol for IoT based on blockchain. They claimed that their protocol is secure against DoS, impersonation, and privileged insider attacks and that it ensures mutual authentication, session key secrecy, and user anonymity. However, their protocol lacks protection against insider, DoS, and stolen verifier attacks, and allows users to be tracked. Therefore, we propose a blockchain-based secure authentication protocol that prevents security attacks, including privileged insider, ephemeral secret leakage, and DoS attacks, while also guaranteeing user anonymity and perfect forward secrecy.

## 3. Preliminaries

In this section, we provide a brief summary of fuzzy extractors and blockchains. The system model and threat model for fog-enabled IoT environments are introduced afterwards.

### *3.1. Fuzzy Extractors*

Fuzzy extraction is an extensively employed method that outputs a fixed value from an input with noise using a helper string [18]. Accordingly, this technique is adopted to generate security parameters from noisy inputs. The biometric information of users, such as face, fingerprints. and irises is a good example of noisy input, as they are not consistent. For successful reproduction, the Hamming distance between the noisy input and registered biometrics should be under a threshold. A fuzzy extractor consists of two algorithms:

- $Gen(Bio) =< \sigma, \tau >$: This probabilistic algorithm is used to generate the biometric secret key $\sigma$ and helper string $\tau$ when biometric data $Bio$ are inserted.
- $Rep(Bio, \tau) = \sigma$: This deterministic algorithm is used to regenerate the original secret key $\sigma$ when $Bio$ and $\tau$ are provided as input. To ensure successful reproduction, the noise mixed in with $Bio$ should be less than the predefined threshold.

Fuzzy extractors can regenerate identical secrets key even when the input value changes by some number of bits, which increases accuracy compared to storing $Bio$ in memory. Fuzzy extractors also require both $Bio$ and $\tau$ in key regeneration, which is more secure than storing $Bio$ in memory.

### *3.2. Blockchain*

Blockchain [42,43] is a distributed ledger technology that allows transactions to be stored in blocks linked together to form a chain. Each block contains information from a previous block along with a timestamp and transaction records. Because each block of the chain contains information from the previous block, blocks cannot be changed once they are added. In addition, all nodes share a copy of the ledger through consensus, which prevents centralization and maintains data quality. Through consensus, blockchains can ensure immutability, as all nodes agree that they are in possession an identical ledger. Consensus algorithms such as Proof-of-Work (PoW) and Proof-of-Stake (PoS) are used to ensure that all network participants agree on a single truth. Widely known blockchains such as Bitcoin [44] and Ethereum [45] use PoW and PoS, respectively. In the proposed protocol, we employ blockchain to enable verification of users in a decentralized method, which can overcome the problems of centralized servers. Moreover, using blockchain can guarantee immutability of uploaded user information, ensuring secure authentication. Among the three types of blockchain (public, private, and consortium), we utilize a public blockchain. This guarantees decentralization, scalability, and transparency, as every node can participate in the consensus process. User credentials are uploaded to the blockchain

for secure user verification; fog nodes retrieve three private user credentials from the blockchain, while the cloud server retrieves the secret user identity.

### *3.3. Threat Model*

In the proposed protocol, we adopt three adversary models: the Dolev–Yao (DY) model [46], Canetti–Krawczyk (CK) model [47], and extended CK (eCK) model [48]. In the DY model, an adversary $A$ can intercept, insert, eavesdrop, and remove messages transmitted over public channels [49]. In the CK model, $A$ can also obtain ephemeral parameters or the private key of the cloud server.

In the eCK model, $A$ can reveal the long-term secret key in order to impersonate a legitimate user. We also assume that $A$ can steal a user's smart device and perform power analysis [13] to extract stored parameters. Using the assumptions of the above three models, $A$ can perform the following attacks:

- An adversary can be disguised as a user, allowing them to be authenticated by other entities. This enables various security attacks using the long-term secret key.
- An adversary can query the blockchain to obtain the user's registration information.
- An adversary can legitimately register and try to acquire the session key of other legitimate users using the collected messages.
- An adversary can perform several types of security attacks, including stolen verifier, MitM, and privileged insider attacks.

### *3.4. System Model*

The proposed system model consists of three layers: the user layer, fog layer, and cloud layer. The details are shown below, and Figure 1 illustrates the proposed system model.

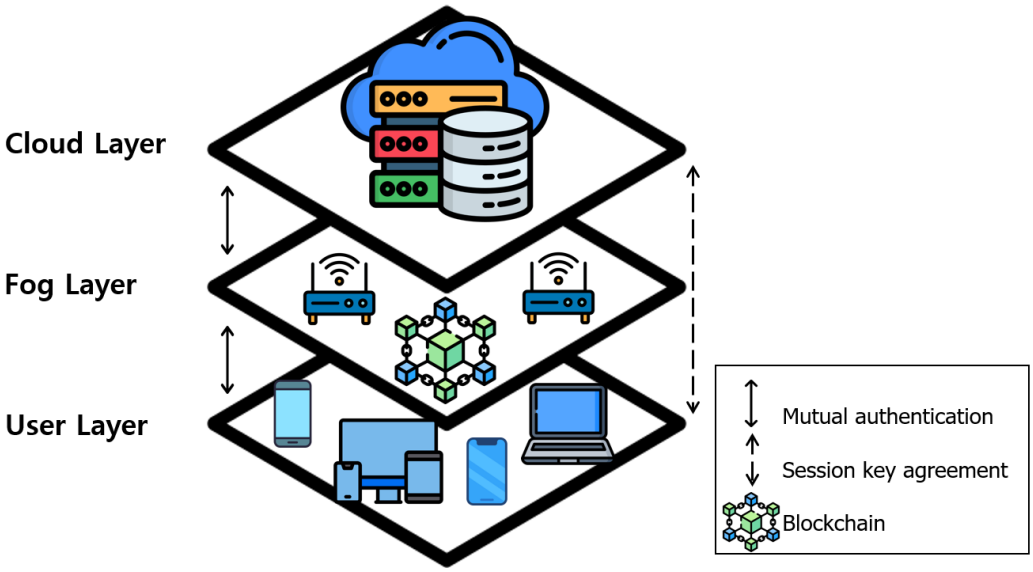


**Figure 1.** Proposed system model.

(1) User layer $U_i$: In this layer, users with resource-constrained smart devices authenticate with the fog layer and the cloud layer to access data produced by IoT devices. Users must register themselves to blockchain-based fog nodes before authentication.

(2) Fog layer $FN_j$: Fog nodes deployed in various areas are connected to the public blockchain in this layer. When users register, their encrypted identifications are uploaded to the blockchain network maintained by fog nodes after consensus process. Verifying users is only available by querying the blockchain in the AKA phase. Fog nodes authenticate users using the blockchain instead of relaying the user's login

request message to the cloud server. Unlike cloud or edge nodes, we adopt fog nodes to overcome problems such as network congestion, single points of failure, and insufficient computing and storage resources.

(3) Cloud layer $CS$: In this layer, the cloud server is a fully trusted entity which stores and processes data collected by IoT devices. The cloud server can query the blockchain for user identification. After authentication, the cloud server provides IoT-based services to users. The server is assumed to have unlimited storage space and computing power.

## 4. Overview of Harbi et al.'s Protocol

In 2024, Harbi et al. [17] proposed a remote user authentication protocol based on blockchain for fog-enabled IoT environments. Their protocol consists of the initialization, user registration, and user login and authentication phases. Table 1 states the notation and descriptions.

**Table 1.** Notation and descriptions

| Notation | Description |
|---|---|
| $SA$ | System administrator |
| $CS$ | Cloud server |
| $FN_j$ | Fog node |
| $U_i$ | User |
| $X$ | Private key of $CS$ |
| $Y$ | Shared secret key between $FN$ and $CS$ |
| $ID_j$ | Identity of $FN_j$ |
| $ID_i$ | Identity of $U_i$ |
| $CF_j$ | Credential of $FN_j$ |
| $TT_i$ | Trust token of $U_i$ |
| $r_k, N_k, \alpha_k$ | Random number |
| $HID_i$ | Pseudo identity of $U_i$ |
| $PID_i$ | Temporary pseudo identity of $U_i$ |
| $h()$ | Hash function |
| $Gen(), Rep()$ | Fuzzy extractor functions |
| $\sigma_i$ | Biometric secret key of $U_i$ |
| $\tau_i$ | Helper string of $U_i$ |
| $T_n$ | Timestamps |
| $\Delta T$ | Maximum transmission delay |
| $SK_i$ | Session key |
| $\oplus, \|\|$ | Exclusive-OR and concatenation operators |

### *4.1. Initialization Phase*

This phase is performed by the system administrator ($SA$).

**Step 1:** $SA$ selects $X$ and $Y$, chooses a unique identity $ID_j$, and computes $CF_j = h(h(ID_j)\|X)$ for each fog node.

**Step 2:** $SA$ stores $(ID_j, CF_j, Y)$ in each fog node and $(h(ID_j), CF_j, Y)$ on the cloud server.

### *4.2. User Registration Phase*

All users must register with the blockchain-based fog nodes to access the cloud server for IoT-associated services. The registered user information is uploaded to the blockchain.

**Step 1:** The user $U_i$ inputs an identity $ID_i$, password $PW_i$, and random number $r_i$.

**Step 2:** $U_i$ calculates $MID_i = h(ID_j \| PW_i)$ and $MPW_i = h(PW_i \| r_i)$. Then, $U_i$ sends $(MID_i, MPW_i)$ to a fog node $FN_j$ through a secure channel.

**Step 3:** $FN_j$ checks the freshness of the received message by querying the blockchain. If it is fresh, $FN_j$ computes $A_i = h(MID_i \| Y), B_i = h(MPW_i \| Y)$ and the user trust token $TT_i = h(A_i \| B_i)$. Its mapping is then created and uploaded to the blockchain. Afterward, $(TT_i, Y)$ are sent back to $U_i$ via a secure channel.

**Step 4:** $U_i$ inputs biometric information $BIO_i$ and computes $Gen(BIO_i) = (\sigma_i, \tau_i), C_i = Y \oplus h(PW_i \| r_i)$, $D_i = h(TT_i \| \sigma_i)$, and $E_i = r_i \oplus h(ID_i \| PW_i)$. Finally, $U_i$ stores $(TT_i, C_i, D_i, E_i, \tau_i)$.

### *4.3. User Login and Authentication Phase*

In this phase, the user, a fog node, and the cloud server carry out the login and authentication phase. Figure 2 depicts the login and authentication phase of Harbi et al.'s protocol. The detailed steps are provided below.

| **User $U_i$** | **Fog Node $FN_j$** | **Cloud server $CS$** |
|---|---|---|
| Input $ID_i, PW_i, BIO_i$ | | |
| Compute $\sigma_i = Rep(BIO_i, \tau_i)$ | | |
| $D_i^* = h(TT_i \| \sigma_i)$ | | |
| Check if $D_i^* \stackrel{?}{=} D_i$ | | |
| Compute $r_i = E_i \oplus h(ID_i \| PW_i)$ | | |
| $MID_i = h(ID_i \| PW_i)$, $MPW_i = h(PW_i \| r_i)$ | | |
| $M_1 = (MID_i \| MPW_i)$ | | |
| Generate timestamp $T_1$ | | |
| $M_2 = h(M_1 \| T_1)$ | | |
| $M_3 = h(M_2) \oplus TT_i$ | | |
| $\xrightarrow{\{M_1, M_2, M_3, T_1\}}$ | | |
| | Check if $\|T_1^* - T_1\| \leq \Delta T$? | |
| | Compute $M_2^* = h(M_1 \| T_1)$, check if $M_2^* = M_2$ | |
| | Compute $TT_i^* = M_3 \oplus h(M_2)$ | |
| | Extract $MID_i$ and $MPW_i$ from $M_1$ | |
| | Verify $(MID_i, MPW_i)$ in Blockchain | |
| | Compare $TT_i^* \stackrel{?}{=} TT_i$ | |
| | Verify mapping $(MID_i, MPW_i, TT_i)$, if validated | |
| | Select $r_j$ | |
| | Compute $M_4 = h(MID_i \| MPW_i \| r_j) \oplus TT_i$ | |
| | Generate timestamp $T_2$ | |
| | Compute $M_5 = h(h(ID_j) \| CF_j \| T_2) \oplus r_j$ | |
| | $\xrightarrow{\{M_1, h(ID_j) M_4, M_5, T_2\}}$ | |
| | | Check if $\|T_2^* - T_2\| \leq \Delta T$? |
| | | Compute $r_j^* = M_5 \oplus h(h(ID_j) \| CF_j \| T_2)$ |
| | | $TT_i^* = M_4 \oplus h(M_1 \| r_j^*)$ |
| | | $M_4^* = h(M_1 \| r_j^* \| TT_i^*)$ |
| | | Check if $M_4^* \stackrel{?}{=} M_4$ |
| | | Select $r_{CS}$ |
| | | compute $SK_i = h(M_1 \| r_{CS} \| TT_i^* \| Y)$ |
| | | Generate a timestamp $T_3$ |
| | | Compute $M_6 = h(M_1 \| Y \| T_3) \oplus r_{CS}$ |
| | | $M_7 = h(SK_i \| M_6)$ |
| | $\xleftarrow{\{M_6, M_7, T_3\}}$ | |
| | Check if $\|T_3^* - T_3\| \leq \Delta T$? | |
| | Compute $M_8 = M_7 \oplus T_4$ | |
| | Generate a timestamp $T_4$ | |
| $\xleftarrow{\{M_6, M_8, T_3, T_4\}}$ | | |
| Check if $\|T_4^* - T_4\| \leq \Delta T$? | | |
| Compute $Y = C_i \oplus h(PW_i \| r_i)$ | | |
| $r_{CS}^* = M_6 \oplus h(M_1 \| Y \| T_3)$ | | |
| $SK_i^* = h(M_1 \| TT_i \| r_{CS} \| Y)$ | | |
| $M_7^* = h(M_6 \| SK_i^*)$ | | |
| $M_8^* = M_7^* \oplus T_4$ | | |
| Check if $M_8^* \stackrel{?}{=} M_8$ | | |
| Use $SK_i$ to secure communication with $CS$ | | |

**Figure 2.** Summary of the login and authentication phase of Harbi et al.'s protocol [17].

**Step 1:** $U_i$ inputs $ID_i, PW_i, BIO_i$ on a smart device. The user's smart device computes $\sigma_i = Rep(BIO_i, \tau_i)$, $D_i^* = h(TT_i \| \sigma_i)$ and verifies $D_i^* \stackrel{?}{=} D_i$. If it is correct, $U_i$ computes $r_i = E_i \oplus h(ID_i \| PW_i)$, $MID_i = h(ID_i \| PW_i)$, and $MPW_i = h(PW_i \| r_i)$. Then, $U_i$ computes $M_1 = (MID_i \| MPW_i)$, generates a timestamp $T_1$, and computes $M_2 = h(M_1 \| T_1)$ and $M_3 = h(M_2) \oplus TT_i$. Finally, $U_i$ sends $\{M_1, M_2, M_3, T_1\}$ to $FN_j$.

**Step 2:** The $FN_j$ receives $\{M_1, M_2, M_3, T_1\}$ and verifies the freshness of $T_1$. If it is fresh, the $FN_j$ computes $M_2^* = h(M_1 \| T_1)$ and checks $M_2^* \stackrel{?}{=} M_2$. If it is equal, $FN_j$ computes $TT_i^* = M_3 \oplus h(M_2)$ and extracts $MID_i$ and $MPW_i$ from $M_1$ to verify them in the blockchain by comparing $TT_i^* \stackrel{?}{=} TT_i$ from the mapping $(MID_i, MPW_i, TT_i)$. If $U_i$ is verified, $U_i$ is authenticated. The $FN_j$ then selects $r_j$, computes $M_4 = h(MID_i \| MPW_i \| r_j) \oplus TT_i$, generates a timestamp $T_2$, and computes $M_5 = h(h(ID_j) \| CF_j \| T_2) \oplus r_j$. Finally, the $FN_j$ sends $\{M_1, h(ID_j), M_4, M_5, T_2\}$ to the cloud server $CS$.

**Step 3:** When the $CS$ receives $\{M_1, h(ID_j), M_4, M_5, T_2\}$ from $FN_j$, the $CS$ first verifies the freshness of $T_2$. Then, the $CS$ computes $r_j^* = M_5 \oplus h(h(ID_j) \| CF_j \| T_2)$, $TT_i^* = M_4 \oplus h(M_1 \| r_j^*)$, and $M_4^* = h(M_1 \| r_j^* \| TT_i^*)$ and checks $M_4^* \stackrel{?}{=} M_4$. If it is equal, the $CS$ authenticates $FN_j$, then selects $r_{CS}$, computes the session key $SK_i = h(M_1 \| r_{CS} \| TT_i^* \| Y)$, generates a timestamp $T_3$, and computes $M_6 = h(M_1 \| Y \| T_3) \oplus r_{CS}$ and $M_7 = h(SK_i \| M_6)$. Finally, the $CS$ sends $\{M_6, M_7, T_3\}$ back to the $FN_j$.

**Step 4:** The $FN_j$ checks the freshness of $T_3$ after receiving $\{M_6, M_7, T_3\}$ from the $CS$. Then, the $FN_j$ generates a timestamp $T_4$, computes $M_8 = M_7 \oplus T_4$, and sends $\{M_6, M_8, T_3, T_4\}$ to the user's smart device.

**Step 5:** After receiving $\{M_6, M_8, T_3, T_4\}$ from the $FN_j$, $U_i$ verifies the freshness of $T_4$. Then, the user's smart device computes $Y = C_i \oplus h(PW_i \| r_i)$, $r_{CS}^* = M_6 \oplus h(M_1 \| Y \| T_3)$, $SK_i^* = h(M_1 \| r_{CS} \| TT_i \| Y)$, $M_7^* = h(SK_i^* \| M_6)$, and $M_8^* = M_7^* \oplus T_4$ and checks $M_8^* \stackrel{?}{=} M_8$. If it is equal, the $FN_j$ is authenticated and $U_i$ can use $SKi$ for secure communication with the $CS$.

## 5. Cryptanalysis of Harbi et al.'s Protocol

In this section, we discuss the security weaknesses of Harbi et al.'s protocol [17] under the DY model and CK model. According to these adversary models, we prove that Harbi et al.'s protocol is vulnerable to insider, stolen verifier, and DoS attacks. We also prove that their protocol does not ensure untraceability of users and has an authentication problem.

### *5.1. Insider Attack*

An adversary $A$ can register with $FN_j$ as a legitimate user and authenticate with $FN_j$ and $CS$ to collect authentication messages. Then, $A$ can invade the session of another user $U_i^l$ and compute the session key $SK_i^l$ of $U_i^l$ using the collected authentication messages. We show this process as follows.

**Step 1:** $A$ inputs its own $ID_i$, $PW_i$, and $BIO_i$ to regenerate $\sigma_i$, compute $D_i = h(TT_i \| \sigma_i)$, and check $D_i^* \stackrel{?}{=} D_i$ for the login process. Then, $A$ computes $M_1 = (MID_i \| MPW_i)$, $M_2 = h(M_1 \| T_1)$, and $M_3 = h(M_2) \oplus TT_i$ to send an authentication request message $\{M_1, M_2, M_3, T_1\}$. After receiving the message $\{M_1, M_2, M_3, T_1\}$, the fog node sends $\{M_1, h(ID_j), M_4, M_5, T_2\}$ to the cloud server, which sends $\{M_6, M_7, T_3\}$ back to the fog node. Finally, the fog node sends $\{M_6, M_8, T_3, T_4\}$ to $A$. Next, $A$ computes the session key $SK_i = h(M_1 \| r_{CS} \| TT_i \| Y)$ and obtains communication messages $\{M_1, M_2, M_3, T_1\}$ and $\{M_6, M_8, T_3, T_4\}$ during the AKA phase. Moreover, $A$ computes $Y = C_i \oplus h(PW_i \| r_i)$ in order to compute the session key of other users.

**Step 2:** $A$ invades the session of $U_i^l$ and intercepts authentication messages $\{M_1^l, M_2^l, M_3^l, T_1^l\}$ and $\{M_6^l, M_8^l, T_3^l, T_4^l\}$. Then $A$ computes $TT_i^l = M_3^l \oplus h(M_2^l)$ and $r_{CS}^l = M_6^l \oplus h(M_1^l \| Y \| T_3^l)$ using $Y$ from $A$'s own authentication phase. Finally, $A$ can compute the session key $SK_i^l$ of the legitimate user $U_i^l$ by computing $SK_i^l = h(M_1^l \| r_{CS}^l \| TT_i^l \| Y)$. Therefore, Harbi et al.'s protocol cannot prevent insider attacks.

*5.2. Stolen Verifier Attack*

If the verification table of the cloud server $\{h(ID_j), CF_j, Y\}$ is leaked to $A$, then $A$ can compute the user's session key. The detailed procedure is described below.

**Step 1:** $A$ obtains authentication messages $\{M_1, M_2, M_3, T_1\}$ and $\{M_6, M_7, T_3\}$ by eavesdropping, then calculates $TT_i = M_3 \oplus h(M_2)$.

**Step 2:** $A$ computes $r_{CS} = M_6 \oplus h(M_1 \| Y \| T_3)$ in order to compute $SK_i = h(M_1 \| r_{CS} \| TT_i \| Y)$. Therefore, Harbi et al.'s protocol is insecure against stolen verifier attacks.

*5.3. Untraceability*

Legitimate users send authentication request messages $\{M_1, M_2, M_3, T_1\}$ containing $M_1 = (MID_i \| MPW_i)$ to fog nodes over public channels. Fog nodes use $MID_i$ and $MPW_i$ in the received $M_1$ to verify users via the mapping $(MID_i, MPW_i, TT_i)$ on the blockchain. Because the values of $ID_i, PW_i$, and $r_i$ do not change in $MID_i = h(ID_i \| PW_i)$ and $MPW_i = h(PW_i \| r_i)$, the user sends an identical $M_1$ in every authentication message. Therefore, $A$ can track the users by checking the value of $M_1$ in the login request message. Thus, Harbi et al.'s protocol cannot ensure user untraceability.

*5.4. DoS Attack*

In this attack, $A$ sends countless authentication request messages to the cloud server via the fog node after obtaining the messages of other users. By performing a DoS attack, $A$ can attempt to prevent legal users from accessing the cloud server by generating large network traffic which can flood the cloud server's resources. The detailed steps are shown below.

**Step 1:** $A$ obtains $\{M_1, M_2, M_3, T_1\}$ from other users and computes $TT_i = M_3 \oplus h(M_2)$. Then, $A$ forges $M_2^A = h(M_1 \| T_1^A)$ and $M_3^A = h(M_2^A) \oplus TT_i$ using $M_1$. Because the value of $M_1$ does not change in every session, the forged $M_2^A$ and $M_3^A$ are valid. $A$ then sends multiple authentication request messages $\{M_1, M_2^A, M_3^A, T_1^A\}$ to the fog node.

**Step 2:** The fog node receives authentication request messages and checks $|T_1^* - T_1^A| \leq \Delta T$. The fog node then proceeds through the user validation process by checking $M_2^* \stackrel{?}{=} M_2^A$, $TT_i^* \stackrel{?}{=} TT_i$ and verifying the mapping $(MID_i, MPW_i, TT_i)$ on the blockchain. As all authentication request messages are forged using legitimate users' credentials, the fog node cannot discard authentication request messages sent by $A$. The fog node continues to follow the steps of the protocol, which eventually becomes a heavy load for the fog node. Therefore, Harbi et al.'s protocol cannot resist DoS attacks.

*5.5. Invalid Authentication Phase*

In the authentication phase, the cloud server authenticates the fog node. When the cloud server receives the message $\{M_1, h(ID_j), M_4, M_5, T_2\}$ from the fog node, the cloud server computes $r_j^* = M_5 \oplus h(h(ID_j \| CF_j \| T_2))$ and $TT_i^* = M_4 \oplus h(M_1 \| r_j)$. Then, the cloud server calculates $M_4^* = h(M_1 \| r_j \| TT_i^*)$ and checks $M_4^* \stackrel{?}{=} M_4$ to authenticate the fog node. However, the value of $M_4$ sent by the fog node is calculated differently, as follows: $M_4 = h(M_1 \| r_j) \oplus TT_i$. The value of $M_4$ cannot be the same; thus, the fog node cannot be authenticated by the cloud server.

## 6. Proposed Protocol

We propose a blockchain-based secure authentication protocol for fog-enabled IoT environments to resolve the security problems of Harbi et al.'s protocol. The proposed pro-

tocol consists of initialization, user registration, login and AKA phase, and user revocation phases. Figure 3 shows the user registration and AKA phases of the proposed protocol.

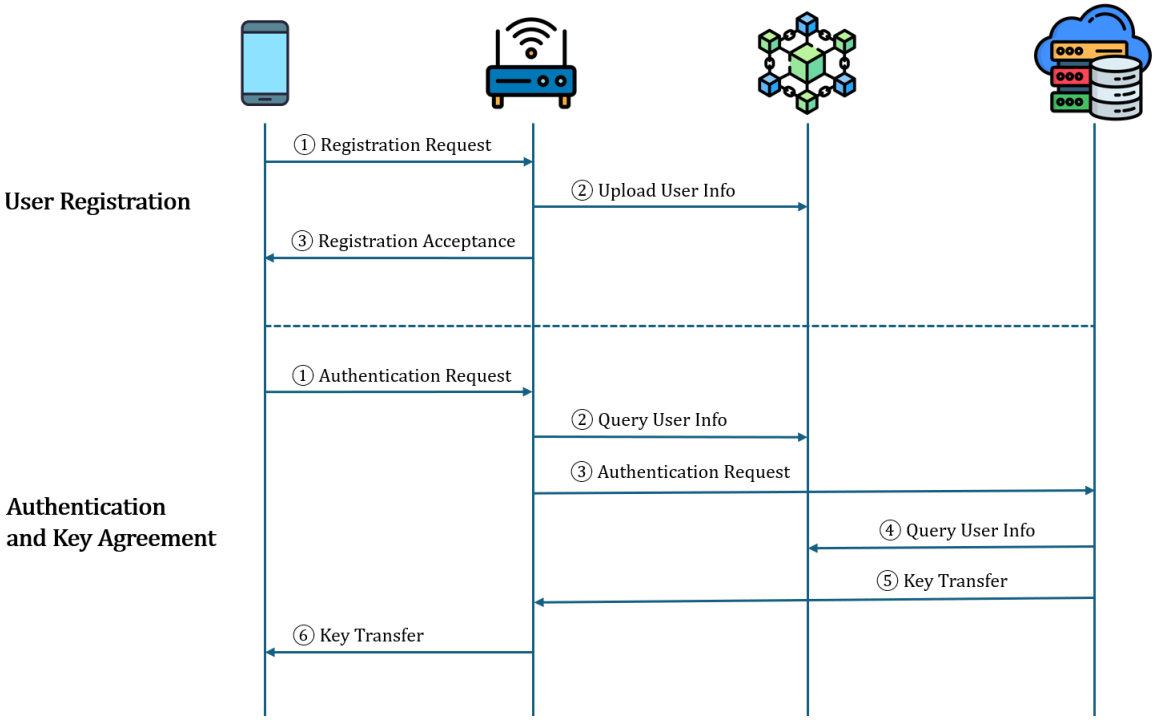


**Figure 3.** Flow of the proposed protocol.

### *6.1. Initialization Phase*

The system administrator $SA$ performs this phase by distributing secret keys, fog node identities, and credentials to the fog nodes and cloud server.

**Step 1:** $SA$ selects a private key $X_{CS}$, a random number $x_{CS}$, and a shared secret key $Y$. Then, $SA$ chooses $ID_j$, computes $CF_j = h(h(ID_j)\|X_{CS})$, and stores $\{ID_j, CF_j, Y\}$ in each fog node.

**Step 2:** Next, $SA$ computes $ID_{j-CS} = h(ID_j) \oplus h(X_{CS}\|x_{CS})$, $CF_{j-CS} = CF_j \oplus h(x_{CS}\|h(ID_j))$, and $Y_{CS} = Y \oplus h(X_{CS}\|h(ID_j)\|CF_j)$. Finally, $SA$ stores $\{ID_{j-CS}, CF_{j-CS}, Y_{CS}, x_{CS}\}$ in the cloud server and concludes the initialization phase.

### *6.2. User Registration Phase*

Users should register with a fog node to access the data stored in the cloud server. The fog node uploads a block containing registered user information to the blockchain for the AKA phase. Using PoW-based blockchain takes 10 min to generate a block on average, while using PoS-based blockchain takes 64 s [50]. After confirming that the block has been uploaded, user authentication is enabled. Figure 4 shows the user registration phase, and the detailed procedures are described below.

**Step 1:** $U_i$ selects an identity $ID_i$ and password $PW_i$, then computes $MID_i = h(ID_i\|PW_i)$, after which $U_i$ sends $\{MID_i\}$ to the nearest fog node.

**Step 2:** When the $FN_j$ receives the message $\{MID_i\}$, it first checks whether $\{MID_i\}$ already exists in the blockchain. If not, the $FN_j$ generates $\alpha_i$ and $HID_i$, then computes $A_i = h(MID_i\|\alpha_i)$, $TT_i = h(A_i\|Y)$, and $PID_i = HID_i \oplus h(Y\|\alpha_i)$. The $FN_j$ then uploads $\{MID_i, A_i, HID_i\}$ to the blockchain and sends $\{A_i, TT_i, PID_i, \alpha_i\}$ back to $U_i$.

**Step 3:** $U_i$ receives the message $\{A_i, TT_i, PID_i, \alpha_i\}$. Then, $U_i$ imprints $BIO_i$ and computes $Gen\ (BIO_i) = < \sigma_i, \tau_i >$, $MPW_i = h(ID_i\|PW_i\|\sigma_i)$, $C_i = MID_i \oplus A_i$, $D_i = MPW_i \oplus$

$TT_i$, and $E_i = h(TT_i \| \sigma_i)$. Fnally, $U_i$ stores $\{\tau_i, C_i, D_i, E_i, PID_i, \alpha_i\}$ in the memory of the user's smart device.

| **User $U_i$** | | **Fog Node $FN_j$** |
|---|---|---|
| Inputs $ID_i, PW_i$ | | |
| Computes $MID_i = h(ID_i \| PW_i)$ | | |
| | $\{MID_i\} \longrightarrow$ | |
| | | Checks $MID_i$ in blockchain |
| | | Generates $\alpha_i, HID_i$ |
| | | Computes $A_i = h(MID_i \| \alpha_i)$ |
| | | $TT_i = h(A_i \| Y)$ |
| | | $PID_i = HID_i \oplus h(Y \| \alpha_i)$ |
| | | Uploads $\{MID_i, A_i, HID_i\}$ to blockchain |
| | $\longleftarrow \{A_i, TT_i, PID_i, \alpha_i\}$ | |
| Imprints $BIO_i$ | | |
| Computes $Gen(BIO_i) =< \sigma_i, \tau_i >$ | | |
| $MPW_i = h(ID_i \| PW_i \| \sigma_i)$ | | |
| $C_i = MID_i \oplus A_i$ | | |
| $D_i = MPW_i \oplus TT_i$ | | |
| $E_i = h(TT_i \| \sigma_i)$ | | |
| Stores $\{\tau_i, C_i, D_i, E_i, PID_i, \alpha_i\}$ | | |

**Figure 4.** User registration phase.

### *6.3. Login and AKA Phase*

Users and the cloud server mutually authenticate each other in open channels with the assistance of fog nodes, using the blockchain to securely share a session key. Fog nodes do not participate in key negotiation, but do perform user verification and transmit messages to users and the cloud server. Fog nodes utilize the registered user information on the blockchain to ensure that the uploaded user information is not manipulated and that the user can be authenticated successfully even if some fog nodes fail. In the AKA phase, the identity of a user $U_i$ is verified by a nearby fog node $FN_j$ by querying the blockchain for user information. After this phase, users can access data stored in the cloud server using the shared session key for secure communication. The proposed protocol's login and AKA phase proceeds as follows, and is shown in Figure 5.

**Step 1:** $U_i$ inputs $ID_i$, $PW_i$, and $BIO_i$. Then, the smart device computes $\sigma_i = Rep(BIO_i, \tau_i)$, $MPW_i = h(ID_i \| PW_i \| \sigma_i)$, $TT_i = MPW_i \oplus D_i$, and $E_i^* = h(TT_i \| \sigma_i)$ and checks $E_i^* \stackrel{?}{=} E_i$. If it holds, the smart device generates $N_u, T_1$ and computes $MID_i = h(ID_i \| PW_i)$, $A_i = C_i \oplus MID_i$, $M_1 = h(N_u \| MPW_i) \oplus h(TT_i \| A_i \| T_1)$, and $M_2 = h(h(N_u \| MPW_i) \| MID_i \| PID_i \| \alpha_i \| T_1)$. Then, $U_i$ sends $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ to $FN_j$.

**Step 2:** When the $FN_j$ receives the user's message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$, the $FN_j$ checks if $|T_1^* - T_1| \leq \Delta T$. If it holds, then the $FN_j$ computes $HID_i = PID_i \oplus h(Y \| \alpha_i)$ and queries $HID_i$ from the blockchain to check whether $HID_i$ exists in the ledger. If it exists, the $FN_j$ extracts $MID_i$ and $A_i$ according to $HID_i$. Then, the $FN_j$ computes $TT_i = h(A_i \| Y)$, $h(N_u \| MPW_i) = M_1 \oplus h(TT_i \| A_i \| T_1)$, and $M_2^* = h(h(N_u \| MPW_i) \| MID_i \| PID_i \| \alpha_i \| T_1)$ and checks whether $M_2^* \stackrel{?}{=} M_2$. If it holds, the $FN_j$ generates $N_F$ and $T_2$, then computes $M_3 = h(h(ID_j) \| CF_j \| T_2) \oplus h(h(N_u \| MPW_i) \| N_F)$, $M_4 = h(h(h(N_u \| MPW_i) \| N_F) \| h(ID_j) \| CF_j \| T_2)$, and $M_5 = MID_i \oplus h(h(h(N_u \| MPW_i) \| N_F) \| CF_j \| T_2)$. Finally, it sends $\{M_3, M_4, M_5, T_2\}$ to the $CS$.

**Step 3:** After the $CS$ receives the message $\{M_3, M_4, M_5, T_2\}$ from the $FN_j$, the $CS$ checks if $|T_2^* - T_2| \leq \Delta T$. If it holds, the $CS$ computes $h(ID_j) = ID_{j-CS} \oplus$

$h(X_{CS}\|x_{CS})$, as the $CS$ can know which fog node sent the message. Then, the $CS$ extracts $CF_{j-CS}$ and $Y_{CS}$ according to $ID_{j-CS}$, computes $CF_j = CF_{j-CS} \oplus h(x_{CS}\|h(ID_j))$, $Y = Y_{CS} \oplus h(X_{CS}\|h(ID_j)\|CF_j)$, $h(h(\ N_u\|MPW_i)\|N_F) = M_3 \oplus h(h(ID_j)\|CF_j\|T_2)$, and $M_4^* = h(h(h(N_u\|MPW_i)\|N_F)\|h(ID_j)\|CF_j\|T_2)$, and checks whether $M_4^* \stackrel{?}{=} M_4$. If the validity is confirmed, the $CS$ generates $N_{CS}$ and $T_3$ and computes $MID_i = M_5 \oplus h(h(h(N_u\ \|MPW_i)\|N_F)\|CF_j\|T_2)$. Next, the $CS$ and queries $A_i$ from the blockchain with $MID_i$. The $CS$ then computes $TT_i = h(A_i\|Y)$, the session key $SK_i = h(TT_i\|h(h(N_u\|MPW_i)\|N_F)\ \|N_{CS})$, $M_6 = h(MID_i\|TT_i\|A_i) \oplus N_{CS}$, $M_7 = h(N_{CS}\|CF_j\|Y\|T_3)$, and $M_8 = h(SK_i\ \|MID_i\|TT_i)$. Finally, the $CS$ sends the message $\{M_6, M_7, M_8, T_3\}$ to the $FN_j$.

**Step 4:** When the $FN_j$ receives the message $\{M_6, M_7, M_8, T_3\}$, the $FN_j$ checks $|T_3^* - T_3| \leq \Delta T$. If it holds, the $FN_j$ computes $N_{CS} = M_6 \oplus h(MID_i\|TT_i\|A_i)$ and $M_7^* = h(N_{CS}\|CF_j\|Y\|T_3)$, then checks $M_7^* \stackrel{?}{=} M_7$. If the two values are same, the $FN_j$ generates a random number $\alpha_{inew}$ and computes $PID_{inew} = HID_i \oplus h(Y\|\alpha_{inew})$, $M_9 = (\alpha_{inew}\|PID_{inew}) \oplus h(h(N_u\|MPW_i)\ \|TT_i\|T_4)$, $M_{10} = h(h(N_u\|MPW_i)\|N_F\|N_{CS}) \oplus h(PID_{inew}\|MID_i\ \|TT_i\ \|T_4)$, and $M_{11} = h(PID_{inew}\|\alpha_{inew}\|M_8\|TT_i\|T_4)$. Then, the $FN_j$ sends $\{M_9, M_{10}, M_{11}, T_4\}$ to $U_i$.

**Step 5:** $U_i$ checks $|T_4^* - T_4| \leq \Delta T$?. If it holds, $U_i$ computes $(\alpha_{inew}\|PID_{inew}) = M_9 \oplus h(h(N_u\|MPW_i)\|TT_i\|T_4)$, $h(h(N_u\|MPW_i)\|N_F\|N_{CS})^* = M_{10} \oplus h(PID_{inew}\|MID_i\ \|TT_i\|T_4)$, $SK_i^* = h(TT_i\|(h(h(N_u\|MPW)\|N_F)\|N_{CS}))^*$, and $M_{11}^* = h(PID_{inew}\|\alpha_{inew}\ \|M_8\ \|TT_i\|T_4)$. Then, $U_i$ verifies $M_{11}^* \stackrel{?}{=} M_{11}$ and updates $\{\alpha_i, PID_i\}$ to $\{\alpha_{inew}, PID_{inew}\}$ in their smart device.

### *6.4. User Device Revocation Phase*

The proposed protocol includes a user device revocation phase to issue new parameters in case the user's device is stolen or lost. This phase proceeds as follows.

**Step 1:** $U_i$ inputs their identity $ID_i^{old}$ and password $PW_i^{old}$ into a new device and computes $MID_i^{old} = h(ID_i^{old}\|PW_i^{old})$. Then, $U_i$ selects a new identity $ID_i^{new}$ and password $PW_i^{new}$ and computes $MID_i^{new} = h(ID_i^{new}\|PW_i^{new})$. Finally, $U_i$ sends $\{MID_i^{old}, MID_i^{new}\}$ to the nearest fog node.

**Step 2:** When the $FN_j$ receives the message $\{MID_i^{old}, MID_i^{new}\}$, the $FN_j$ first checks for the existence of $\{MID_i^{old}\}$ in the blockchain. Then, the $FN_j$ updates $\{MID_i^{old}\}$ to $\{MID_i^{new}\}$, generates $\alpha_i^{new}$ and $HID_i^{new}$, and computes $A_i^{new} = h(MID_i^{new}\|\alpha_i^{new})$, $TT_i^{new} = h(A_i^{new}\|Y)$, and $PID_i^{new} = HID_i^{new} \oplus h(Y\|\alpha_i^{new})$. The $FN_j$ uploads the updated user credentials $\{MID_i^{new}, A_i^{new}, HID_i^{new}\}$ to the blockchain and sends $\{A_i^{new}, TT_i^{new}, PID_i^{new}, \alpha_i^{new}\}$ back to $U_i$. In future authentications, the $FN_j$ queries the updated user credentials for user verification.

**Step 3:** After receiving the message $\{A_i^{new}, TT_i^{new}, PID_i^{new}, \alpha_i^{new}\}$, $U_i$ imprints $BIO_i$ and computes $Gen(BIO_i) = < \sigma_i, \tau_i >$, $MPW_i = h(ID_i^{new}\|PW_i^{new}\|\sigma_i)$, $C_i^{new} = MID_i^{new} \oplus A_i^{new}$, $D_i^{new} = MPW_i^{new} \oplus TT_i^{new}$, and $E_i^{new} = h(TT_i^{new}\|\sigma_i)$. Then, $U_i$ stores $\{\tau_i, C_i^{new}, D_i^{new}, E_i^{new}, PID_i^{new}, \alpha_i^{new}\}$ in the memory of the user's smart device.

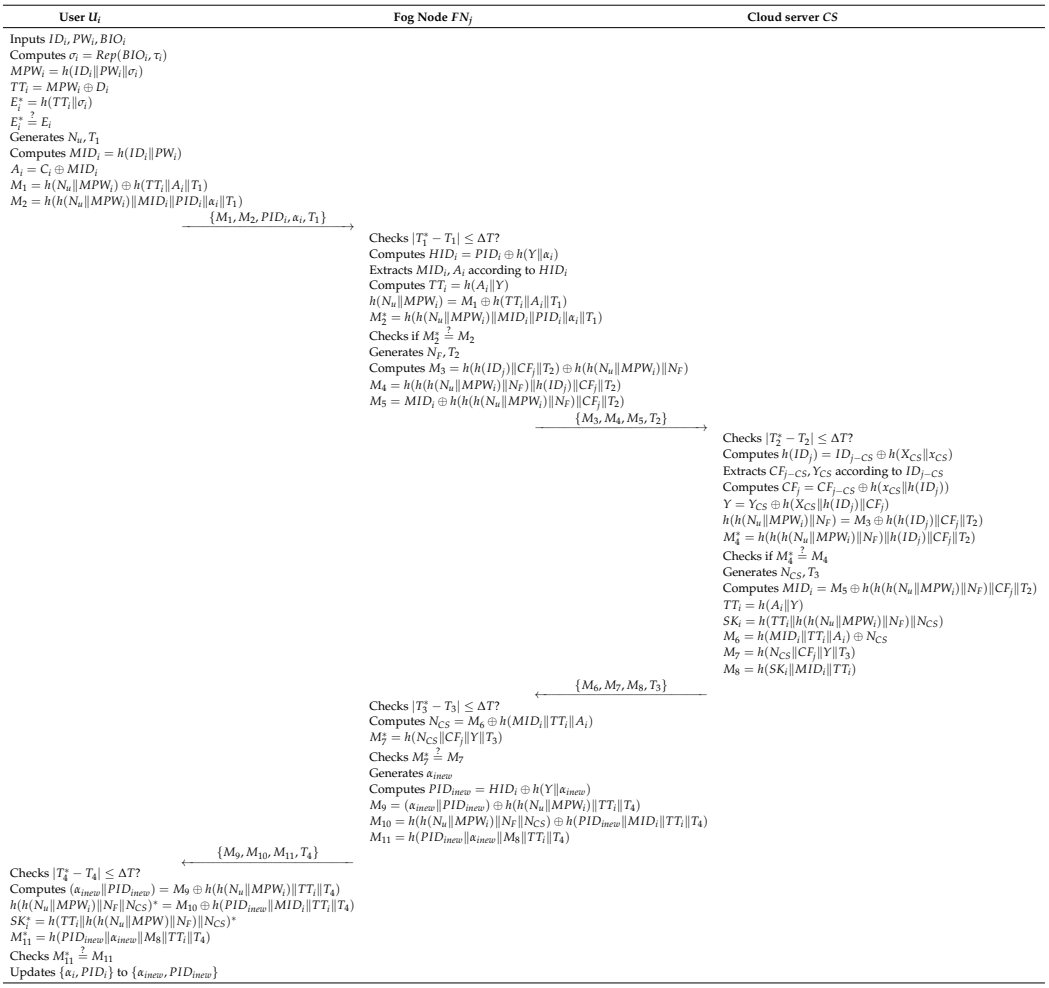


**Figure 5.** Login and AKA phase.

## 7. Security Analysis

The security features of the proposed protocol are analyzed using formal and informal methods. BAN logic, the RoR model, and AVISPA are utilized for formal analysis, and informal analyses are also conducted to prove our protocol's security.

### *7.1. BAN Logic*

The widely known BAN logic [19] is a formal proof for verification of of a protocol's mutual authentication. We prove that the proposed protocol ensures mutual authentication via BAN logic. Table 2 provides the notation and descriptions.

#### 7.1.1. Rules

The basic rules of BAN logic are listed below.

**1.** Message Meaning Rule (MMR):

$$\frac{N_1 \mid \equiv N_1 \overset{K}{\leftrightarrow} N_2, \quad N_1 \lhd \{S_1\}_K}{N_1 \mid \equiv N_2 \mid \sim S_1}$$

**2.** Nonce Verification Rule (NVR):

$$\frac{N_1 \mid \equiv \#(S_1), \quad N_1 \mid \equiv N_2 \mid \sim S_1}{N_1 \mid \equiv N_2 \mid \equiv S_1}$$

**3.** Jurisdiction Rule (JR):

$$\frac{N_1| \equiv N_2| \Longrightarrow S_1, \quad N_1| \equiv N_2| \equiv S_1}{N_1 \Big| \equiv S_1}$$

**4.** Belief Rule (BR):

$$\frac{N_1 \Big| \equiv (S_1, S_2)}{N_1 \Big| \equiv S_1}$$

**5.** Freshness Rule (FR):

$$\frac{N_1 \Big| \equiv \#(S_1)}{N_1 \Big| \equiv \#(S_1, S_2)}$$

**Table 2.** BAN logic notation.

| Notation | Description |
|---|---|
| $N_1, N_2$ | Principals |
| $S_1, S_2$ | Statements |
| $N_1\| \equiv S_1$ | $N_1$ **believes** $S_1$ |
| $N_1\| \sim S_1$ | $N_1$ once **said** $S_1$ |
| $N_1 \Rightarrow S_1$ | $N_1$ **controls** $S_1$ |
| $N_1 \lhd S_1$ | $N_1$ **receives** $S_1$ |
| $\#S_1$ | $S_1$ is **fresh** |
| $\{S_1\}_K$ | $S_1$ is **encrypted** with $K$ |
| $N_1 \overset{K}{\longleftrightarrow} N_2$ | $N_1$ and $N_2$ have **shared** key $K$ |

7.1.2. Goals

In our paper, we define the user as $U$, the fog node as $FN_j$, and the cloud server as $CS$. $U$ and $CS$ authenticate each other by establishing a session key $SK_i$. To prove that $SK_i$ is shared only between $U$ and $CS$, the following four goals should be achieved.

**Goal 1:** $U| \equiv U \overset{SK_i}{\longleftrightarrow} CS$
**Goal 2:** $U| \equiv CS| \equiv U \overset{SK_i}{\longleftrightarrow} CS$
**Goal 3:** $CS| \equiv U \overset{SK_i}{\longleftrightarrow} CS$
**Goal 4:** $CS| \equiv U| \equiv U \overset{SK_i}{\longleftrightarrow} CS$

7.1.3. Idealized Forms

In the proposed protocol, network participants send four messages $\{M_1, M_2, PID_i, \alpha_i, T_1\}$, $\{M_3, M_4, M_5, T_2\}$, $\{M_6, M_7, M_8, T_3\}$, and $\{M_9, M_{10}, M_{11}, T_4\}$ over insecure channels. These messages need to be transformed into idealized forms for BAN logic analysis, as shown below.

**Message 1:** $U \rightarrow FN_j : \{h(N_u \| MPW_i)\}_{TT_i}$
**Message 2:** $FN_j \rightarrow CS : \{h(h(N_u \| MPW_i) \| N_F)\}_{CF_j}$
**Message 3:** $FN_j \leftarrow CS : \{N_{CS}\}_{CF_j}$
**Message 4:** $U \leftarrow FN_j : \{h(h(N_u \| MPW_i) N_F \| N_{CS})\}_{TT_i}$

7.1.4. Assumptions

The BAN logic assumptions of the proposed protocol are listed below. The network participants believe that they have received fresh timestamps and are sharing keys for

secure communication. The following assumptions are required in order to meet the aforementioned goals.

$A_1$: $FN_j| \equiv \#(T_1)$
$A_2$: $CS| \equiv \#(T_2)$
$A_3$: $FN_j| \equiv \#(T_3)$
$A_4$: $U| \equiv \#(T_4)$
$A_5$: $U| \equiv U \overset{TT_i}{\longleftrightarrow} FN_j$
$A_6$: $FN_j| \equiv U \overset{TT_i}{\longleftrightarrow} FN_j$
$A_7$: $U| \equiv CS \Rightarrow (U \overset{SK_i}{\longleftrightarrow} CS)$
$A_8$: $CS| \equiv U \Rightarrow (U \overset{SK_i}{\longleftrightarrow} CS)$
$A_9$: $CS| \equiv CS \overset{CF_j}{\longleftrightarrow} FN_j$
$A_{10}$: $FN_j| \equiv CS \overset{CF_j}{\longleftrightarrow} FN_j$

7.1.5. BAN Logic Proof

We prove that the proposed protocol ensures mutual authentication between the network participants using rules, idealized forms, and assumptions. The four goals specified in Section 7.1.2 are achieved in the following steps.

**Step 1:** $S_1$ can be obtained from $Msg_1$.

$$S_1 : FN_j \lhd \{h(N_u \| MPW_i), T_1\}_{TT_i}$$

**Step 2:** $S_2$ can be obtained by applying $MMR$ to $S_1$ and $A_6$.

$$S_2 : FN_j| \equiv U| \sim (h(N_u \| MPW_i), T_1)$$

**Step 3:** $S_3$ can be obtained by applying $FR$ to $S_2$ and $A_1$.

$$S_3 : FN_j| \equiv \#(h(N_u \| MPW_i), T_1)$$

**Step 4:** $S_4$ can be obtained by applying $NVR$ to $S_2$ and $S_3$.

$$S_4 : FN_j| \equiv U| \equiv (h(N_u \| MPW_i), T_1)$$

**Step 5:** $S_5$ can be obtained from $Msg_2$.

$$S_5 : CS \lhd \{h(h(N_u \| MPW_i) \| N_F), T_2\}_{CF_j}$$

**Step 6:** $S_6$ can be obtained by applying $MMR$ to $S_5$ and $A_9$.

$$S_6 : CS| \equiv FN_j| \sim (h(h(N_u \| MPW_i) \| N_F), T_2)$$

**Step 7:** $S_7$ can be obtained by applying $FR$ to $S_6$ and $A_2$.

$$S_7 : CS| \equiv \#(h(h(N_u \| MPW_i) \| N_F), T_2)$$

**Step 8:** $S_8$ can be obtained by applying $NVR$ to $S_6$ and $S_7$.

$$S_8 : CS| \equiv FN_j| \equiv (h(h(N_u \| MPW_i) \| N_F), T_2)$$

**Step 9:** $S_9$ can be obtained from $Msg_3$.

$$S_9 : FN_j \lhd \{N_{CS}, T_3\}_{CF_j}$$

**Step 10:** $S_{10}$ can be obtained by applying $MMR$ to $S_9$ and $A_{10}$.

$$S_{10} : FN_j| \equiv FN_j| \sim (N_{CS}, T_3)$$

**Step 11:** $S_{11}$ can be obtained by applying $FR$ to $S_{10}$ and $A_3$.

$$S_{11} : FN_j| \equiv \#(N_{CS}, T_3)$$

**Step 12:** $S_{12}$ can be obtained by applying $NVR$ to $S_{10}$ and $S_{11}$.

$$S_{12} : FN_j| \equiv CS| \equiv (N_{CS}, T_3)$$

**Step 13:** $S_{13}$ can be obtained from $Msg_4$.

$$S_{13} : U \lhd \{h(h(n_u \| MPW_i) \| N_F \| N_{CS}), T_4\}_{TT_i}$$

**Step 14:** $S_{14}$ can be obtained by applying $MMR$ to $S_{13}$ and $A_5$.

$$S_{14} : U| \equiv FN_j| \sim (h(h(N_u \| MPW_i) \| N_F \| N_{CS}), T_4)$$

**Step 15:** $S_{15}$ can be obtained by applying $FR$ using $S_{14}$ to $A_4$.

$$S_{15} : U| \equiv \#(h(h(N_u \| MPW_i) \| N_F \| N_{CS}), T_4)$$

**Step 16:** $S_{16}$ can be obtained by applying $NVR$ to $S_{14}$ and $S_{15}$.

$$S_{16} : U| \equiv FN_j| \equiv (h(h(N_u \| MPW_i) \| N_F \| N_{CS}), T_4)$$

**Step 17:** $S_{17}$ and $S_{18}$ can be obtained using $S_4$, $S_8$, $S_{12}$, and $S_{16}$ to achieve Goal 2 and Goal 4. Through the four steps, parts of $SK_i$ are passed through $FN_j$, which has been proven to be trustworthy from the past steps. The $CS$ receives $(h(h(N_u \| MPW_i) \| N_F)$ and $U_i$ receives $(h(h(N_u \| MPW_i) \| N_F \| N_{CS})$ to compute $SK_i = h(TT_i \| h(h(N_u \| MPW_i) \| N_F) \| N_{CS})$.

$$S_{17} : U| \equiv CS| \equiv U \overset{SK_i}{\longleftrightarrow} CS \textbf{ (Goal 2)}$$

$$S_{18} : CS| \equiv U| \equiv U \overset{SK_i}{\longleftrightarrow} CS \textbf{ (Goal 4)}$$

**Step 18:** $S_{19}$ can be obtained by applying $JR$ to $S_{17}$ and $A_7$, while $S_{20}$ can be obtained by applying $JR$ to $S_{18}$ and $A_8$, as shown below.

$$S_{19} : U| \equiv U \overset{SK_i}{\longleftrightarrow} CS \textbf{ (Goal 1)}$$

$$S_{18} : CS| \equiv U \overset{SK_i}{\longleftrightarrow} CS \textbf{ (Goal 3)}$$

Therefore, the four BAN logic goals are achieved, proving that the proposed protocol ensures mutual authentication.

### *7.2. RoR Model*

We confirm the session key security of the proposed protocol using the RoR model [20]. The proposed protocol has three participants: a user $p_U^{t_1}$, fog node $p_{FN_j}^{t_2}$, and cloud server $p_{CS}^{t_3}$. In the RoR model, we assume that the adversary $A$ can use the transmitted public messages to perform attacks through various queries, such as *Execute*, *CorruptSD*, *Send*, and *Test*. We present the details of the security proof following the steps in [51,52]. The queries and their definitions are as follows:

- $Execute(P_U^{t_1}, P_{FN_j}^{t_2}, P_{CS}^{t_3})$: $A$ can eavesdrop on messages transmitted over the public channels between the three legitimate entities. $A$ can attempt various security attacks based on the obtained messages. The *Execute* query is a passive attack.
- $CorruptSD(P_U^{t_1})$: $A$ can obtain all stored secret parameters from the smart device of the legitimate user $P_U^{t_1}$. Thus, the *CorruptSD* query is an active attack.

- $Send(p^t, Msg)$: $A$ can transmit a message to network participants $p^t$ and receive a response of the message according to the protocol. The *Send* query has the attributes of an active attack.
- $Test(p^t)$: $A$ flips a coin $C$ and receives a random string or a session key. If the session key $SK_i$ is fresh, then $A$ obtains $C = 1$; if not, then $C = 0$. Otherwise, $A$ obtains a null value ($\perp$). If $A$ is not able to distinguish the result, then $SK_i$ is secure. $A$ can perform *Test* queries as many times as desired.

**Theorem 1.** *The adversary $A$ attempts to calculate the session key $SK_i$. Thus, we define $Adv_A$ as the advantage of $A$ for this model. Here, $q_{hash}$ and $q_{send}$ are the number of Hash and Send queries executed by $A$, respectively, with $|Hash|$ as the range of Hash, while $C$ and $s$ are Zipf's parameters [53]. Let $Adv_A$ be the advantage of $A$ for this model.*

$$Adv_A \leq \frac{q_{hash}^2}{|Hash|} + 2max\{C \cdot q_{send}^s, \frac{q_{send}}{2^{l_D}}\} \tag{1}$$

**Proof.** $A$ performs various attacks in the following games: $G_n$ $(n = 0, 1, 2, 3)$. $Pr[Succ_n]$ is the probability of $A$ guessing $c$ correctly in $G_n$.

$G_0$: In $G_0$, $A$ does not perform any queries and has no information about $SK_i$. $A$ can only guess the random bit $C$.

$$Adv_A = |2Pr[Succ_0] - 1|. \tag{2}$$

$G_1$: In $G_1$, $A$ performs the *Execute* and *Test* queries to verify the obtained session key. $A$ tries to calculate the session key $SK_i = h(TT_i \| h(h(N_u \| MPW_i) \| N_F) \| N_{CS})$ with the obtained public messages. However, $A$ must acquire $MPW_i$, $TT_i$, and the three random nonces $N_u$, $N_F$, and $N_{CS}$ in order to calculate $SK_i$. However, $A$ cannot compute the necessary parameters from the obtained public messages; therefore, $A$ can obtain no advantage from $G_1$.

$$Pr[Succ_1] = Pr[Succ_0] \tag{3}$$

$G_2$: In $G_2$, $A$ performs *Send* and *Hash* queries to acquire the session key. However, all messages obtained by *Send* queries use the one-way hash function $h(.)$ to prevent leakage of any useful information. To compute the session key, $A$ must forge a valid message that collides with the obtained message by using *Hash* queries. Hence, utilizing the birthday paradox [54], we can induce the Equation (4).

Proof of Equation (4): Let $P_i$ be the event where the $i$-th *Hash* query collides with one of the previous queries. Then, $Pr[P_i] \leq \frac{i-1}{|Hash|}$, because the $i$-th query just needs to be the same as one of the queries. We also let $P(|Hash|, q_h)$ denote the possibility of an occurrence of at least one collision when performing $q_h$ in $|Hash|$. Then, $Pr[Succ_2] - \frac{1}{2} = P(|Hash|, q_h) = P_1 + P_2 + \cdots + P_{q_{hash}} = \frac{0}{|Hash|} + \frac{1}{|Hash|} + ... + \frac{q_h - 1}{|Hash|} = \frac{q_{hash}(q_{hash}-1)}{2|Hash|} \leq \frac{q_{hash}^2}{2|Hash|}$.

$$|Pr[Succ_2] - Pr[Succ_1]| \leq \frac{q_{hash}^2}{2|Hash|} \tag{4}$$

$G_3$: Finally, in $G_3$, $A$ executes the *CorruptSD* query to obtain stored parameters $\{\tau_i, C_i, D_i, E_i, PID_i, \alpha_i\}$ from the smart device of $U$, where $C_i = MID_i \oplus A_i$, $D_i = MPW_i \oplus TT_i$, $E_i = h(TT_i \| \sigma_i)$, $\tau_i$ denotes a helper string, $PID_i$ represents a one-time pseudo-identifier, and $\alpha_i$ indicates a random nonce. To legitimately send a login request message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ to the fog node, $A$ must guess the value of $ID_i$, $PW_i$, and $BIO_i$, which is impossible in polynomial time. By Zipf's law [53], we can derive Equation (5).

Proof of Equation (5): Zipf's law explains the relationship between frequency and rank in natural language. The frequency of a word is inversely proportionate to its rank in the frequency table, listed in decreasing order. According to the authors of [53], Zipf's law can be applied to the distribution of passwords in real life. Their CDF-Zipf model is $F_r = C^{'} \cdot r^{s^{'}}$, where $F_r$ is the cumulative frequency of passwords up to rank $r$, $C^{'}$ and where $s^{'}$ are constants for the password dataset. Because $r = 1, 2, 3, \cdots, q_{send}$ and because $A$ can use the dictionary to guess the password starting from rank 1 and below, or can just guess the password with the length of $l_D$ bits, $A$ has the possibility of guessing the correct password $Pr[Succ_3] - \frac{1}{2} = max\{C \cdot q_{send}^s, \frac{q_{send}}{2^{l_D}}\}$.

$$|Pr[Succ_3] - Pr[Succ_2]| \leq max\{C \cdot q_{send}^s, \frac{q_{send}}{2^{l_D}}\} \tag{5}$$

When the games are all over, $A$ guesses bit $c$. From the past four games, no information about bit $C$ is leaked to $A$. Therefore, we can obtain Equation (6).

$$Pr[Succ_3] = \frac{1}{2} \tag{6}$$

Using Equations (2) and (3), we can obtain Equation (7).

$$\frac{1}{2}Adv_A = |Pr[Succ_0] - \frac{1}{2}| = |Pr[Succ_1] - \frac{1}{2}| \tag{7}$$

Moreover, using (6) and (7) yields Equation (8).

$$\frac{1}{2}Adv_A = |Pr[Succ_1] - Pr[Succ_3]| \tag{8}$$

Applying the triangular inequality provides the following Equation (9).

$$\begin{aligned} \frac{1}{2}Adv_A &= |Pr[Succ_1] - Pr[Succ_3]| \\ &\leq |Pr[Succ_1] - Pr[Succ_2]| \\ &+ |Pr[Succ_2] - Pr[Succ_3]| \\ &\leq max\{C \cdot q_{send}^s, \frac{q_{send}}{2^{l_D}}\} \end{aligned} \tag{9}$$

By multiplying two on both sides of (9), we finally obtain the desired result (10).

$$Adv_A \leq \frac{q_{hash}^2}{|Hash|} + 2max\{C \cdot q_{send}^s, \frac{q_{send}}{2^{l_D}}\}. \tag{10}$$

□

### *7.3. AVISPA Simulation*

We simulate the proposed protocol using AVISPA [21,22], which is a security tool that can detect replay and MitM attacks. For AVISPA simulation, we code the High-Level Protocol Specification Language (HLPSL) to simulate the proposed protocol. The HLPSL code is converted to the Intermediate Format (IF), then the IF is put into four back-ends: the On-the-Fly Model Checker (OFMC), Constraint Logic-based Attack Searcher (CL-AtSe), SAT-based Model Checker (SATMC), and Three Automata based on Automatic Approximations for Analysis of Security Protocol (TA4SP). The OFMC and CL-AtSe back-ends are selected because our protocol includes exclusive-OR operations. The protocol is considered to be secure against replay and MitM attacks if a "SAFE" message is obtained in the Output Format (OF).

Below, we provide an explanation of the HLPSL code for the proposed protocol. The three entities $U_i$, $FN_j$, and $CS$ in our protocol are written as *roles*. Figure 6 presents the HLPSL code of the session and environment roles in our protocol.

```
role session(A:agent, B:agent, C:agent, H:hash_func)
def=
local
SND3, RCV3, SND2, RCV2, SND1, RCV1: channel(dy)
composition
userdevice(A,B,C,H, SND1,RCV1)
/\ fognode(A,B,C,H, SND2, RCV2)
/\ cloudserver(A,B,C,H, SND3, RCV3)
end role

role environment()
def=
const
a,b,c: agent,
h:hash_func,
pidi, alphai, t1,t2,t3,t4,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,ski:text,
sec1,sec2,sec3, auth1, auth2, auth3:protocol_id
intruder_knowledge = {a,b,c,h,pidi,alphai,t1,t2,t3,t4,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11}
composition                %110
session(a,b,c,h)
/\ session(i,b,c,h)
/\ session(a,i,c,h)

end role

goal
        secrecy_of sec1, sec2, sec3
        authentication_on auth1
        authentication_on auth2
        authentication_on auth3
end goal

environment()
```

**Figure 6.** HLPSL code of the session, environment, and goals.

Figure 7 shows the role of the user. In state 1, $U_i$ generates $N_u$ and $T_1$, then computes $MID_i$, $A_i$, and $MPW_i$. Here, $secret(ID_i, PW_i, sec1, A)$ means $ID_i$ and $PW_i$ is a secret only known to $U_i$. Then, $U_i$ sends $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ to $FN_j$. Later in state 2, $U_i$ receives $\{M_9, M_{10}, M_{11}, T_4\}$ and verifies the session key and $M_{11}$. $U_i$ performs $request(C, A, auth3, N_{CS})$ to accept the freshness of $N_{CS}$.

```
role userdevice(A:agent, B:agent, C:agent, H:hash_func, SND, RCV:channel(dy))
played_by A
def=
local
State:nat,
IDi, PWi, Bioi, TTi, MIDi, MPWi, Nu, Nf, Ncs,
Y,Ci, Ai, Sigmai, PIDi, Alphai, Var, T1, T3, T4, M1, M2, M3, M6, M7, M8, M9, M10, M11, SKi: text
const      sec1, sec2, sec3, auth1, auth2, auth3:protocol_id
init
State := 0
transition
1. State=0 /\ RCV(start) =|>
State' :=1
/\ Nu' := new()
/\ T1' := new()
/\ MIDi' :=H(IDi. PWi)
/\ Ai' := xor(Ci, MIDi')
/\ MPWi' := H(IDi.PWi. Sigmai)
/\ M1' :=xor(H(Nu'.MPWi'), H(TTi.Ai.T1'))
/\ M2' := H(H(Nu'.MPWi').MIDi'.PIDi.Alphai.T1' )
/\ SND(M1'. M2'.PIDi.Alphai. T1')                    %%21
/\ secret({IDi, PWi}, sec1, A)
/\ witness(A, B, auth1, TTi)
2. State=1 /\ RCV(M9'. M10'. M11'. T4') =|>
State' :=2
/\ Var' :=xor(M9', H(H(Nu.MPWi).TTi.T4))
/\ SKi' :=H(TTi.H(H(Nu.H(IDi.PWi.Sigmai)).Nf).Ncs)
/\ M11':=H(Var'.H(H(TTi.H(H(Nu.H(IDi.PWi.Sigmai)).Nf).Ncs).H(IDi.PWi).TTi).TTi.T4')
/\ request(C, A, auth3, Ncs)
end role
```

**Figure 7.** HLPSL code of the user.

The simulation results using the OFMC and CL-AtSe back-ends are provided in Figure 8. The summary indicates that $A$ cannot conduct replay or MitM attacks against the proposed protocol.

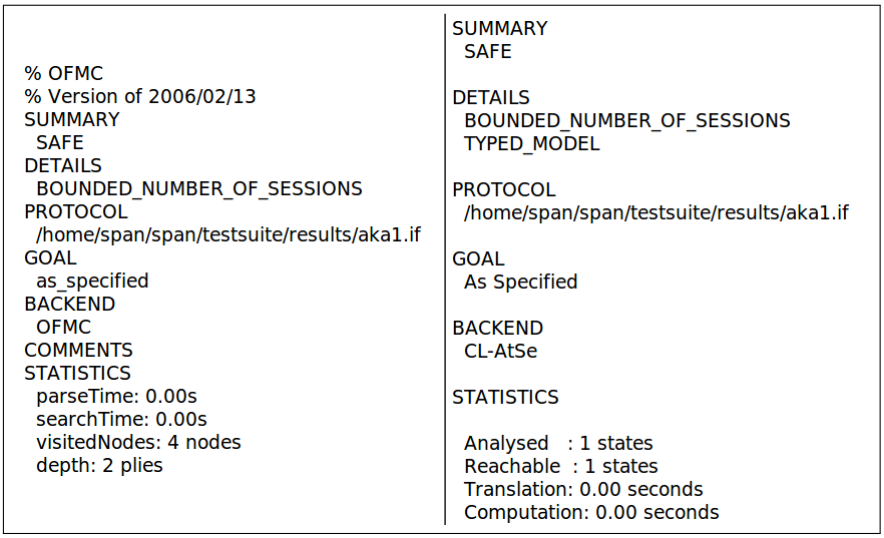


**Figure 8.** AVISPA simulation results.

### *7.4. Informal Analysis*

Formal analyses such as the RoR model, BAN logic, and AVISPA can verify security vulnerabilities, session key security, and mutual authentication. However, these analyses have difficulties in proving potential security attacks. Thus, we conduct informal analysis to verify the security robustness of the proposed protocol, including physical, stolen verifier, DoS, and desynchronization attacks.

#### 7.4.1. Impersonation Attacks

To impersonate a user, fog node, or cloud server, $A$ needs to create messages to send to other network entities. Specifically, $A$ must calculate valid $A_i$ and $TT_i$ to impersonate a user. Moreover, $A$ requires $Y$, $h(ID_j)$, and $CF_j$, which are securely stored in the fog node and cloud server, in order to impersonate the fog node or cloud server. Thus, our protocol can prevent impersonation attacks.

#### 7.4.2. Insider Attacks

First, $A$ registers with the $FN_j$ as a user and authenticates with the $FN_j$ and $CS$. Afterwards, $A$ can invade other sessions to compute the session keys of other users. However, $SK_i$ consists of random numbers which are different in every session. Moreover, a user's $TT_i$ cannot be calculated without $A_i$ and $Y$. Therefore, the proposed protocol can defend against insider attacks.

#### 7.4.3. Off-Line Guessing Attacks

$A$ obtains a user's smart device and extracts the secret parameters $\{\tau_i, C_i, D_i, E_i, HID_i\}$ by power analysis. Afterwards, $A$ tries to guess the correct $PW_i$ using the extracted parameters. $A$ needs to either guess $MID_i = h(ID_i \| PW_i)$ using $MID_i = C_i \oplus A_i$ or guess $MPW_i = h(ID_i \| PW_i \| \sigma_i)$ using $MPW_i = D_i \oplus TT_i$. However, $A_i$ and $TT_i$ are values that the fog node sends to the user, and are not stored. Furthermore, to impersonate the legitimate user, $A$ needs $MPW_i$ in order to calculate $TT_i = D_i \oplus MPW_i$ for the user authentication request message. $MPW_i$ contains not only $ID_i$ and $PW_i$ but also $\sigma_i$, which makes guessing $MPW_i$ computationally infeasible. Moreover, $\sigma_i = Rep(BIO_i, \tau_i)$ is the biometric secret key of a user generated by the fuzzy extractor. Hence, the proposed protocol can resist offline guessing attacks.

7.4.4. Physical Attacks

$A$ can perform physical attacks such as side-channel attacks or smart device capture. In this way, $A$ obtains secret parameters $\{\tau_i, C_i, D_i, E_i, HID_i\}$; however, without $TT_i$ and $MPW_i$, impersonating users or cracking $SK_i$ is not possible. Moreover, even if $A$ steals the smart device and extracts secret parameters, the legitimate user can receive new parameters by following the procedure in Section 6.4, making the stolen parameters useless. Therefore, the proposed protocol can defend against physical attacks.

7.4.5. Replay and MitM Attacks

In our protocol, random nonces and timestamps are included in every message and change in every session. When network participants receive a message, they check the freshness of the message using timestamps. Moreover, every message is masked with secret parameters that are shared offline. Thus, the proposed protocol can defend against replay and MitM attacks.

7.4.6. DoS Attacks

$A$ can send numerous requests targeting the fog nodes in order to interfere with the login and authentication process. In the proposed protocol, calculating a login request message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ requires knowledge of the user's secret parameters, such as $MPW_i$ and $TT_i$. If $A$ sends many messages, the fog node can filter the received messages by checking $|T_1^* - T_1| \leq \Delta T$. Even if $T_1$ is perceived as a valid timestamp, the fog node can calculate $M_2$ to filter the message. Because $M_2$ contains secret parameters shared between $U_i$ and $FN_j$, $A$ cannot calculate the correct value of $M_2$. Hence, the proposed protocol can resist DoS attacks.

7.4.7. Mutual Authentication and Session Key Security

In the proposed protocol, all messages include timestamps $T_1$, $T_2$, $T_3$, and $T_4$ to determine the freshness of the messages, and network participants validate the integrity of the messages by checking $M_2$, $M_4$, $M_7$, and $M_{11}$. For the security of the session key, we have proved that $A$ cannot calculate a valid session key according to Sections 7.4.12 and 7.4.13. Hence, the proposed protocol guarantees mutual authentication and session key security.

7.4.8. Untraceability

In the AKA phase of the proposed protocol, the fog node verifies the user by calculating the pseudo-identity of $U_i$, that is, $HID_i = PID_i \oplus h(Y \| \alpha_i)$. In every session, the user sends $PID_i$ and $\alpha_i$ in the authentication request message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ to the fog node. In the authentication response message $\{M_9, M_{10}, M_{11}, T_4\}$, the fog node sends dynamically updated pseudo-parameters $(\alpha_{inew} \| PID_{inew})$ masked with $h(h(N_u \| MPW_i) \| TT_i \| T_4)$ to the user for a future session and to prevent tracking by $A$. Moreover, even if $A$ intercepts $\alpha_i$ and $PID_i$ from the public message, $A$ still cannot compute $HID_i$, as $A$ does not know the value of $Y$. Therefore, the proposed protocol guarantees user untraceability.

7.4.9. Desynchronization Attacks

The user sends $PID_i$ and $\alpha_i$ to the fog node at the beginning of authentication phase. The fog node then sends $PID_{inew}$ and $\alpha_{inew}$ to the user. $A$ can attempt to delay the messages to interrupt authentication, but $HID_i$ can be recovered by any matching $PID_i$ and $\alpha_i$. The fog node can recover $HID_i$ even if the user sends the $PID_i$ and $\alpha_i$ again. Thus, the proposed protocol can prevent desynchronization attacks.

7.4.10. Privileged Insider Attacks

$A$ obtains the user registration request message $\{MID_i\}$. $A$ also steals the smart device of a legitimate user to perform power analysis to extract stored parameters $\{\tau_i, C_i, D_i, E_i, HID_i\}$. However, $A$ cannot impersonate the user, because $A$ cannot calculate $TT_i = MPW_i \oplus D_i$ without knowing the correct $ID_i$, $PW_i$, and $\sigma_i$. Hence, the proposed protocol can resist privileged insider attacks.

7.4.11. Perfect Forward Secrecy

Suppose that $A$ obtains the private key $X_{CS}$. In this case, no useful information about the $SK_i$ can be derived from $X_{CS}$. In the proposed protocol, the session key contains three random nonces, $N_u$, $N_F$, and $N_{CS}$, each created using the three network participants and secret parameters of $U_i$. Thus, the proposed protocol ensures perfect forward secrecy.

7.4.12. Session-Specific Random Number Leakage Attacks

Assume that $A$ obtains three random numbers $N_u$, $N_F$, and $N_{CS}$ along with the public messages. In this case, $A$ cannot compute $SK_i$, as they are unable to calculate $TT_i = h(A_i \| Y)$ and $MPW_i = h(ID_i \| PW_i \| \sigma_i)$. Hence, the proposed protocol can resist session-specific random number leakage attacks.

7.4.13. Stolen Verifier Attacks

Assume that the verification table $\{ID_{j-CS}, CF_{j-CS}, Y_{CS}, x_{CS}\}$ of the cloud server is leaked to $A$. Without knowing the private key $X_{CS}$, $A$ cannot calculate $h(ID_j) = ID_{j-CS} \oplus h(X_{CS} \| x_{CS})$, $CF_j = CF_{j-CS} \oplus h(x_{CS} \| h(ID_j))$, and $Y = Y_{CS} \oplus h(X_{CS} \| h(ID_j) \| CF_j)$ to calculate the session key $SK_i$ or impersonate as the fog node or cloud server. Therefore, the proposed protocol can prevent stolen verifier attacks.

7.4.14. Privacy Preservation and User Anonymity

$A$ attempts to calculate the real identity $ID_i$ of the user $U_i$ using public messages. However, all messages are masked by random nonces and secret parameters using hash functions. Moreover, $\alpha_i$ and $PID_i$ are random nonces that are renewed every session, and contain no information about the user identity. Without knowledge of $Y$, it is difficult for $A$ to compute $HID_i = PID_i \oplus h(Y \| \alpha_i)$ in order to determine which user is requesting authentication. It is also hard for $A$ to decide whether or not two users in different sessions are the same user. Only the fog nodes can verify the user by making queries according to the uploaded user credentials $\{MID_i, A_i, HID_i\}$ in the blockchain, and these queries are not recorded. Therefore, the proposed protocol preserves privacy and ensures user anonymity.

7.4.15. Key Compromise Impersonation (KCI) Attack

$A$ acquires the long-term secret parameters $\{Y, X_{CS}\}$. However, $A$ cannot compute valid user authentication request message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ without the user information $MID_i$, $A_i$, and $TT_i$. $A$ cannot calculate $MID_i = h(ID_i \| PW_i)$ without knowing $ID_i$ and $PW_i$, and cannot calculate $A_i = h(MID_i \| \alpha_i)$ without knowing both $MID_i$ and $\alpha_i$. Thus, the proposed protocol can resist KCI attacks.

## 8. Performance Analysis

We compare the proposed protocol with other studies [17,30–35], analyzing the computation and communication costs as well as security and functionality properties. We also conduct simulation using NS-3 considering real-world applications.

### *8.1. Security Features Comparison*

We compare the security and the functionality features of the proposed protocol with other related protocols [17,30–35]. We define (SF 1: "Impersonation attack"), (SF 2: "Insider attack"), (SF 3: "Offline guessing attack"), (SF 4: "Stolen user device, smart card"), (SF 5: "Replay attack"), (SF 6: "MitM attack"), (SF 7: "DoS attack"), (SF 8: "Mutual authentication and session key security"), (SF 9: "User untraceability"), (SF 10: "Desynchronization attack"), (SF 11: "Privileged insider attack"), (SF 12: "Forward secrecy"), (SF 13: "Session-specific random number leakage attack"), (SF 14: "Stolen verifier attack"), (SF 15: "User anonymity"), and (SF 16: "Key Compromise Impersonation attack"), respectively. The security features and functionalities are shown in Table 3. Except for the two protocols by Qiao et al. and Tomar et al., which have no parameter update process for SF 10, all features are compared. Moreover, other protocols do not adopt the eCK model and do not have protection against KCI attacks. As shown, the proposed protocol provides more security and functionality features than other related protocols. Therefore, the proposed protocol provides necessary security features for fog-enabled IoT environments.

**Table 3.** Comparison of security and functionality features.

| Features | Smet et al. [30] | Saleem et al. [31] | Tahir et al. [32] | Qiao et al. [33] | Irshad et al. [34] | Tomar et al. [35] | Harbi et al. [17] | Proposed |
|---|---|---|---|---|---|---|---|---|
| **SF 1** | × | ○ | ○ | × | ○ | × | ○ | ○ |
| **SF 2** | ○ | ○ | ○ | × | ○ | × | × | ○ |
| **SF 3** | ○ | ○ | ○ | ○ | ○ | × | ○ | ○ |
| **SF 4** | × | ○ | × | ○ | ○ | × | ○ | ○ |
| **SF 5** | ○ | × | × | ○ | ○ | ○ | ○ | ○ |
| **SF 6** | ○ | ○ | ○ | × | ○ | ○ | ○ | ○ |
| **SF 7** | ○ | ○ | ○ | × | × | × | × | ○ |
| **SF 8** | ○ | ○ | ○ | × | ○ | ○ | ○ | ○ |
| **SF 9** | ○ | × | × | × | ○ | ○ | × | ○ |
| **SF 10** | × | ○ | ○ | − | ○ | − | ○ | ○ |
| **SF 11** | ○ | ○ | × | ○ | ○ | ○ | ○ | ○ |
| **SF 12** | ○ | × | × | ○ | ○ | ○ | ○ | ○ |
| **SF 13** | ○ | ○ | × | × | × | ○ | ○ | ○ |
| **SF 14** | ○ | ○ | × | ○ | ○ | ○ | × | ○ |
| **SF 15** | × | ○ | × | ○ | ○ | ○ | ○ | ○ |
| **SF 16** | × | ○ | × | × | ○ | × | × | ○ |

○: "Provided security or functionality feature"; ×: "not provided security or functionality feature"; −: "Not considered".

### *8.2. Communication Cost Comparison*

This section presents an analysis of the communication overhead generated during the authentication phase. According to [31,33], we set the bit sizes of the hash digest, identity, random number, timestamp, PUF response, Chebyshev polynomial, ECC point, and symmetric encryption to 160, 128, 128, 32, 128, 160, 320, and 128 bits, respectively. With this information, the communication costs of the proposed protocol can be calculated. The comparison results of the communication costs of related studies, including the protocol of Harbi et al. [17,30–35], are also shown in Table 4.

- Message 1: The message $\{M_1, M_2, PID_i, \alpha_i, T_1\}$ needs (160 + 160 + 128 + 128 + 32) = 608 bits.
- Message 2: The message $\{M_3, M_4, M_5, T_2\}$ needs (160 + 160 + 160 + 32) = 512 bits.

- Message 3: The message $\{M_6, M_7, M_8, T_3\}$ requires (128 + 160 + 160 + 32) = 480 bits.
- Message 4: The message $\{M_9, M_{10}, M_{11}, T_4\}$ requires (256 + 160 + 160 + 32) = 608 bits.

The total communication costs of the proposed protocol are 608 + 512 + 480 + 608 = 2208 bits. The protocols by [30–34] do not utilize blockchain, meaning that the message from the end device needs to be relayed to the central server for authentication. In [35], Tomar et al. use ECC, where the ECC point has a size of 320 bits. In [17], $M_1$ is repeatedly transmitted for authentication. Therefore, the proposed protocol has lower communication costs than other related protocols, including that of Harbi et al. [17,30–33,35].

**Table 4.** Comparison of communication costs between the proposed protocol and related works.

| Protocols | Total Costs (bits) | Messages |
|---|---|---|
| Smet et al. [30] | 4800 | 8 |
| Saleem et al. [31] | 3552 | 4 |
| Tahir et al. [32] | 2656 | 4 |
| Qiao et al. [33] | 3744 | 4 |
| Irshad et al. [34] | 4640 | 4 |
| Tomar et al. [35] | 4192 | 4 |
| Harbi et al. [17] | 2240 | 4 |
| Proposed | 2208 | 4 |

### *8.3. Computational Cost Comparison*

In this section, we compute the computational costs of the proposed protocol in comparison to other related protocols. According to [17,55], we define the hash function (≈0.0023 ms), fuzzy extractor (≈2.226 ms), ECC scalar multiplication (≈2.226 ms), symmetric encryption and decryption (≈0.0046 ms), and ECC addition (≈0.288 ms) as $T_H$, $T_F$, $T_{EM}$, $T_{SYM}$, and $T_{EA}$, respectively. For PUF operation time (≈0.12 ms), we follow the result from [56] and denote it as $T_{PUF}$. In [57], the execution time for random number generation is specified as (≈0.539 ms, $T_{RNG}$), while [39] states that the operation time for chaotic mapping is one third that of ECC scalar multiplication (≈0.742 ms, $T_{CM}$). Table 5 indicates the computational cost of the proposed protocol in comparison to other related protocols. The computational cost of Harbi et al.'s protocol [17] is slightly lower than that of the proposed protocol. However, our proposed protocol provides better security features than that of Harbi et al. Moreover, our proposed protocol is more lightweight compared to other protocols [33–35] that use ECC and Chebyshev polynomial. The protocols presented by [30–32] use lightweight cryptographic operations, as in our proposed protocol, but also perform more time-consuming operations such as fuzzy extraction and random number generation.

### *8.4. NS-3 Simulation*

We simulated the proposed protocol using the NS-3 open-source discrete-event network simulator [23]. NS-3 can be used to simulate the data flows in a real-world network. With NS-3, we examined the end-to-end delay and throughput of the AKA phase in the proposed protocol. The length of each message was set as follows: $\{M_1, M_2, PID_i, \alpha_i, T_1\}$: 76 bytes; $\{M_3, M_4, M_5, T_2\}$: 64 bytes; $\{M_6, M_7, M_8, T_3\}$: 60 bytes; $\{M_9, M_{10}, M_{11}, T_4\}$: 76 bytes. The number of users was set to 10, 20, 30, 40, and 50. The fog nodes and cloud server consisted of fixed infrastructure that communicates with as many users as possible. We performed the simulation using Ubuntu 16.04 LTS on an Intel Core i5-11400 @ 2.60 GHz CPU with 24.0 GB RAM. The simulation parameters are shown in Table 6.

**Table 5.** Comparison of computational costs.

| Protocol | End Device | Infrastructure | Total Costs (ms) |
|---|---|---|---|
| Smet et al. [30] | $4T_{SYM} + 2T_{PUF} + 1T_F + 12T_H + 1T_{RNG}$ | $10T_{SYM} + 2T_{PUF} + 2T_F + 27T_H + 2T_{RNG}$ | 8.9291 |
| Saleem et al. [31] | $1T_{SYM} + 1T_{PUF} + 2T_F + 6T_H + 1T_{RNG}$ | $2T_{SYM} + 1T_{PUF} + 1T_F + 10T_H + 2T_{RNG}$ | 8.581 |
| Tahir et al. [32] | $1T_F + 3T_H + 2T_{RNG}$ | $1T_{SYM} + 10T_H + 3T_{RNG}$ | 4.9555 |
| Qiao et al. [33] | $2T_{CM} + 7T_H + 1T_{RNG}$ | $6T_{CM} + 4T_{SYM} + 17T_H + 4T_{RNG}$ | 8.7046 |
| Irshad et al. [34] | $2T_{CM} + 1T_{SYM} + 8T_H + 1T_{RNG}$ | $6T_{CM} + 6T_{SYM} 16T_H + 5T_{RNG}$ | 9.2574 |
| Tomar et al. [35] | $4T_{EM} + 7T_H + 1T_{RNG}$ | $13T_{EM} + 4T_{EA} + 15T_H + 2T_{RNG}$ | 39.6248 |
| Harbi et al. [17] | $1T_F + 8T_H$ | $11T_H + 2T_{RNG}$ | 3.3477 |
| Proposed | $1T_F + 11T_H + 1T_{RNG}$ | $27T_H + 2T_{RNG}$ | 3.9304 |

In the proposed protocol, end-to-end delay and throughput were calculated as follows: end-to-end delay $= \sum_{i=1}^{A_t} \frac{A_{rcv} - A_{snd}}{A_t}$; throughput $= \frac{P_{rcv} * P_{size}}{T}$, where $A_t$, $A_{rcv}$, $A_{snd}$, $P_r$, $P_s$, and $T$ denote the total number of authentication message packets, received time, sent time, total packets received, packet size, and total time, respectively. The simulation results are shown in Figure 9. The results demonstrate a proportionate increase in end-to-end delay and throughput as the number of users increases.

**Table 6.** Parameters for the NS-3 simulation.

| Simulation Parameters | Description |
|---|---|
| RAM specification | Samsung DDR4 2666 MHz 24.0 GB |
| Operating systems | Ubuntu 16.04 LTS |
| CPU specification | Intel Core i5-11400 @ 2.60 GHz |
| NS-3 version | 3.29 |
| Mobility model | RandomDirection2dMobilityModel ConstantPositionMobilityModel |
| Propagation loss model | TwoRayGroundPropagationLossModel |
| Routing protocol | Ad-hoc On-demand Distance Vector |
| Network | 802.11ac |
| Simulation area | 500 × 500 $m^2$ |
| Simulation time | 500 s |
| Number of users | 10, 20, 30, 40, 50 |

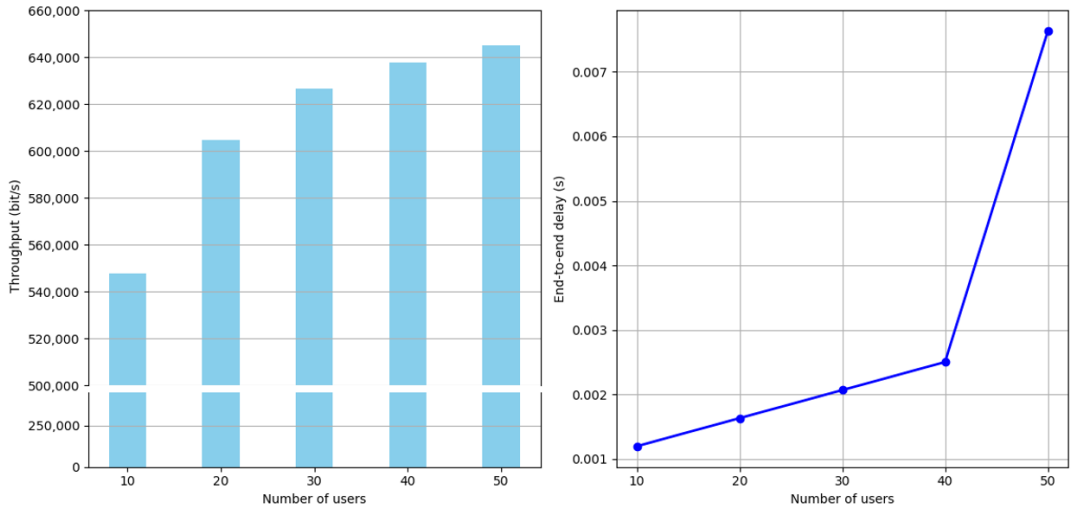


**Figure 9.** Throughput and end-to-end delay of our NS-3 simulation.

## 9. Conclusions

In this paper, we have reviewed Harbi et al.'s protocol and proved that their protocol can allow insider, stolen verifier, and DoS attacks. We have also discovered that their protocol cannot ensure user untraceability and has an authentication problem. To address these security problems, we propose a blockchain-based secure authentication protocol for fog-enabled IoT environments. We confirm that the proposed protocol provides security features via AVISPA, the RoR model, and BAN logic. We also confirm by informal analysis that our protocol can resist various security attacks. Finally, we compare the performance of our protocol with related protocols through comparative analysis and conduct simulation for practical deployment using NS-3. Unlike authentication protocols that use ECC, the proposed protocol uses lightweight cryptographic operations; despite this, it can defend against more diverse security attacks. Moreover, the proposed protocol adopts three threat models, namely, the DY model, CK model, and eCK model, demonstrating high robustness against more complex attacks. The results of our performance analysis reveal that the proposed protocol is more suitable for fog-enabled IoT environments than other protocols in terms of both efficiency and security. In future work, we plan to devise a more suitable authentication protocol for fog-enabled IoT environments.

**Author Contributions:** Conceptualization, T.K.; methodology, T.K. and D.K.; validation, Y.P. (Yohan Park) and Y.P. (Youngho Park); formal analysis, T.K. and D.K.; writing—original draft preparation, T.K.; writing—review and editing, D.K. and Y.P. (Yohan Park); supervision, Y.P. (Yohan Park) and Y.P. (Youngho Park); project administration, Y.P. (Youngho Park); funding acquisition, Y.P. (Yohan Park). All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (Ministry of Science and ICT) (RS-2024-00450915).

**Data Availability Statement:** The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding authors.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Ayaz, M.; Ammad-Uddin, M.; Sharif, Z.; Mansour, A.; Aggoune, E.H.M. Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk. *IEEE Access* **2019**, *7*, 129551–129583. [CrossRef]
2. Sutrala, A.K.; Obaidat, M.S.; Saha, S.; Das, A.K.; Alazab, M.; Park, Y. Authenticated key agreement scheme with user anonymity and untraceability for 5G-enabled softwarized industrial cyber-physical systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 2316–2330. [CrossRef]
3. Reddi, S.; Rao, P.M.; Saraswathi, P.; Jangirala, S.; Das, A.K.; Jamal, S.S.; Park, Y. Privacy-preserving electronic medical record sharing for IoT-enabled healthcare system using fully homomorphic encryption, IOTA, and masked authenticated messaging. *IEEE Trans. Ind. Inform.* **2024**, *20*, 10802–10813. [CrossRef]
4. Botta, A.; De Donato, W.; Persico, V.; Pescapé, A. Integration of cloud computing and internet of things: A survey. *Future Gener. Comput. Syst.* **2016**, *56*, 684–700. [CrossRef]
5. Fox, G.C.; Kamburugamuve, S.; Hartman, R.D. Architecture and measured characteristics of a cloud based internet of things. In Proceedings of the International Conference on Collaboration Technologies and Systems (CTS), Denver, CO, USA, 21–25 May 2012; pp. 6–12.
6. Atlam, H.F.; Walters, R.J.; Wills, G.B. Fog computing and the internet of things: A review. *Big Data Cogn. Comput.* **2018**, *2*, 10. [CrossRef]
7. Gunawi, H.S.; Hao, M.; Suminto, R.O.; Laksono, A.; Satria, A.D.; Adityatama, J.; Eliazar, K.J. Why does the cloud stop computing? lessons from hundreds of service outages. In Proceedings of the Seventh ACM Symposium on Cloud Computing, Santa Clara, CA, USA, 5–7 October 2016; pp. 1–16.
8. Mouradian, C.; Naboulsi, D.; Yangui, S.; Glitho, R.H.; Morrow, M.J.; Polakos, P.A. A comprehensive survey on fog computing: State-of-the-art and research challenges. *IEEE Commun. Surv. Tutor.* **2017**, *20*, 416–464. [CrossRef]

9. Al Faruque, M.A.; Vatanparvar, K. Energy management-as-a-service over fog computing platform. *IEEE Internet Things J.* **2015**, *3*, 161–169. [CrossRef]
10. Peter, N. Fog computing and its real time applications. *Int. J. Emerg. Technol. Adv. Eng.* **2015**, *5*, 266–269.
11. Zhang, J.; Fang, H.; Zhong, H.; Cui, J.; He, D. Blockchain-assisted privacy-preserving traffic route management scheme for fog-based vehicular ad-hoc networks. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 2854–2868. [CrossRef]
12. Kumari, A.; Tanwar, S.; Tyagi, S.; Kumar, N. Fog computing for Healthcare 4.0 environment: Opportunities and challenges. *Comput. Electr. Eng.* **2018**, *72*, 1–13. [CrossRef]
13. Kocher, P.; Jaffe, J.; Jun, B. Differential power analysis. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 15–19 August 1999; pp. 388–397.
14. Bicakci, K.; Tavli, B. Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks. *Comput. Stand. Interfaces* **2009**, *31*, 931–941. [CrossRef]
15. Almadhoun, R.; Kadadha, M.; Alhemeiri, M.; Alshehhi, M.; Salah, K. A User Authentication Scheme of IoT Devices using Blockchain-Enabled Fog Nodes. In Proceedings of the 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA), Aqaba, Jordan, 28 October–1 November 2018; pp. 1–8.
16. Guo, Y.; Zhang, Z.; Guo, Y.; Xiong, P. BSRA: Blockchain-based secure remote authentication scheme for fog-enabled Internet of Things. *IEEE Internet Things J.* **2023**, *11*, 3348–3361. [CrossRef]
17. Harbi, Y.; Aliouat, Z.; Harous, S.; Gueroui, A.M. Lightweight blockchain-based remote user authentication for fog-enabled IoT deployment. *Comput. Commun.* **2024**, *221*, 90–105. [CrossRef]
18. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptology-EUROCRYPT 2004, Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 523–540.
19. Burrows, M.; Abadi, M.; Needham, R. A logic of authentication. *ACM Trans. Comput. Syst. (TOCS)* **1990**, *8*, 18–36. [CrossRef]
20. Abdalla, M.; Fouque, P.; Pointcheval, D. Password-based authenticated key exchange in the three-party setting. In *Public Key Cryptography—PKC 2005, Proceedings of the 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, 23–26 January 2005*; Lecture Notes in Computer Science (LNCS); Springer: Berlin/Heidelberg, Germany, 2005; pp. 65–84.
21. AIVSPA. Automated Validation of Internet Security Protocols and Applications. Available online: https://avispa-project.org/main (accessed on 24 April 2025).
22. SPAN: A Security Protocol Animator for AIVSPA. Available online: https://people.irisa.fr/Thomas.Genet/span/ (accessed on 24 April 2025).
23. Network Simulator 3. Available online: https://www.nsnam.org (accessed on 10 June 2025).
24. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
25. Hazra, A.; Rana, P.; Adhikari, M.; Amgoth, T. Fog computing for next-generation internet of things: Fundamental, state-of-the-art and research challenges. *Comput. Sci. Rev.* **2023**, *48*, 100549. [CrossRef]
26. Burhan, M.; Alam, H.; Arsalan, A.; Rehman, R.A.; Anwar, M.; Faheem, M.; Ashraf, M. . A comprehensive survey on the cooperation of fog computing paradigm-based IoT applications: Layered architecture, real-time security issues, and solutions. *IEEE Access* **2023**, *11*, 73303–73329. [CrossRef]
27. Jia, X.; He, D.; Kumar, N.; Choo, K.K.R. Authenticated key agreement scheme for fog-driven IoT healthcare system. *Wirel. Netw.* **2019**, *25*, 4737–4750. [CrossRef]
28. Ma, M.; He, D.; Wang, H.; Kumar, N.; Choo, K.K.R. An efficient and provably secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks. *IEEE Internet Things J.* **2019**, *6*, 8065–8075. [CrossRef]
29. Eftekhari, S.A.; Nikooghadam, M.; Rafighi, M. Security-enhanced three-party pairwise secret key agreement protocol for fog-based vehicular ad-hoc communications. *Veh. Commun.* **2021**, *28*, 100306–100322. [CrossRef]
30. De Smet, R.; Vandervelden, T.; Steenhaut, K.; Braeken, A. Lightweight PUF based authentication scheme for fog architecture. *Wirel. Netw.* **2021**, *27*, 947–959. [CrossRef]
31. Saleem, M.A.; Li, X.; Ayub, M.F.; Shamshad, S.; Wu, F.; Abbas, H. An efficient and physically secure privacy-preserving key-agreement protocol for vehicular ad-hoc network. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 9940–9951. [CrossRef]
32. Tahir, H.; Mahmood, K.; Ayub, M.F.; Saleem, M.A.; Ferzund, J.; Kumar, N. Lightweight and secure multi-factor authentication scheme in VANETs. *IEEE Trans. Veh. Technol.* **2023**, *72*, 14978–14986. [CrossRef]
33. Qiao, H.; Dong, X.; Jiang, Q.; Ma, S.; Liu, C.; Xi, N.; Shen, Y. Anonymous lightweight authenticated key agreement protocol for fog-assisted healthcare IoT system. *IEEE Internet Things J.* **2023**, *10*, 16715–16726. [CrossRef]
34. Irshad, A.; Aljaedi, A.; Bassfar, Z.; Jamal, S.S.; Usman, M.; Chaudhry, S.A. FA-SMW: Fog-driven anonymous lightweight access control for smart medical wearables. *IEEE Internet Things J.* **2024**, *12*, 4275–4285. [CrossRef]

35. Tomar, A.; Tripathi, S. Blockchain-assisted authentication and key agreement scheme for fog-based smart grid. *Clust. Comput.* **2022**, *25*, 451–468. [CrossRef]
36. Subramani, J.; Maria, A.; Rajasekaran, A.S.; Al-Turjman, F.; Gopal, M. Blockchain-based physically secure and privacy-aware anonymous authentication scheme for fog-based vanets. *IEEE Access* **2022**, *11*, 17138–17150. [CrossRef]
37. Ravi, B.; Kumar, M.; Hu, Y.C.; Hassan, S.; Kumar, B. Stochastic modeling and performance analysis in balancing load and traffic for vehicular ad hoc networks: A review. *Int. J. Netw. Manag.* **2023**, *33*, e2224. [CrossRef]
38. Wei, L.; Cui, J.; Zhong, H.; Bolodurina, I.; Liu, L. A lightweight and conditional privacy-preserving authenticated key agreement scheme with multi-TA model for fog-based VANETs. *IEEE Trans. Dependable Secur. Comput.* **2021**, *20*, 422–436. [CrossRef]
39. Tomar, A.; Tripathi, S. A Chebyshev polynomial-based authentication scheme using blockchain technology for fog-based vehicular network. *IEEE Trans. Mob. Comput.* **2024**, *23*, 9075–9089. [CrossRef]
40. Subramani, J.; Maria, A.; Sivaraman, A.; Vijayakumar, P.; Alqahtani, F.; Tolba, A. An efficient anonymous authentication scheme for blockchain assisted and fog-enabled smart grid. *Comput. Electr. Eng.* **2024**, *119*, 109508–109522. [CrossRef]
41. Alsaeed, N.; Nadeem, F.; Albalwy, F. A scalable and lightweight group authentication framework for Internet of Medical Things using integrated blockchain and fog computing. *Future Gener. Comput. Syst.* **2024**, *151*, 162–181. [CrossRef]
42. Son, S.; Lee, J.; Park, Y.; Park, Y.; Das, A.K. Design of blockchain-based lightweight V2I handover authentication protocol for VANET. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 1346–1358. [CrossRef]
43. Ryu, J.; Son, S.; Lee, J.; Park, Y.; Park, Y. Design of secure mutual authentication scheme for metaverse environments using blockchain. *IEEE Access* **2022**, *10*, 98944–98958. [CrossRef]
44. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Bus. Rev.* **2008**, 21260. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 11 June 2025).
45. Buterin, V. A Next-Generation Smart Contract and Decentralized Application Platform. Available online: https://ethereum.org/en/whitepaper/ (accessed on 11 June 2025).
46. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [CrossRef]
47. Canetti, R.; Krawczyk, H. Universally composable notions of key exchange and secure channels. In *Advances in Cryptology—EUROCRYPT 2002, Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, 28 April–2 May 2002*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 337–351.
48. LaMacchia, B.; Lauter, K.; Mityagin, A. Stronger security of authenticated key exchange. In *International Conference on Provable Security, Berlin, Heidelberg, November 2007*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 1–16.
49. Wazid, M.; Bagga, P.; Das, A.K.; Shetty, S.; Rodrigues, J.J.; Park, Y. AKM-IoV: Authenticated key management protocol in fog computing-based Internet of vehicles deployment. *IEEE Internet Things J.* **2019**, *6*, 8804–8817. [CrossRef]
50. Yang, F.; Zhou, W.; Wu, Q.; Long, R.; Xiong, N.N.; Zhou, M. Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism. *IEEE Access* **2019**, *7*, 118541–118555. [CrossRef]
51. Kwon, D.; Son, S.; Kim, M.; Lee, J.; Das, A.K.; Park, Y. A secure self-certified broadcast authentication protocol for intelligent transportation systems in UAV-assisted mobile edge computing environments. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 19004–19017. [CrossRef]
52. Yu, S.; Park, Y. A robust authentication protocol for wireless medical sensor networks using blockchain and physically unclonable functions. *IEEE Internet Things J.* **2022**, *9*, 20214–20228. [CrossRef]
53. Wang, D.; Cheng, H.; Wang, P.; Huang, X.; Jian, G. Zipf's law in passwords. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 2776–2791. [CrossRef]
54. Boyko, V.; MacKenzie, P.; Patel, S. Provably secure password-authenticated key exchange using Diffie-Hellman. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Bruges, Belgium, 14–18 May 2000; pp. 156–171.
55. Harbi, Y.; Aliouat, Z.; Refoufi, A.; Harous, S.; Bentaleb, A. Enhanced authentication and key management scheme for securing data transmission in the internet of things. *Ad Hoc Netw.* **2019**, *94*, 101948–101961. [CrossRef]
56. Gope, P. PMAKE: Privacy-aware multi-factor authenticated key establishment scheme for advance metering infrastructure in smart grid. *Comput. Commun.* **2020**, *152*, 338–344. [CrossRef]
57. Kilinc, H.H.; Yanik, T. A survey of SIP authentication and key agreement schemes. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 1005–1023. [CrossRef]