applied sciences

MDPI

Article

2FAKA-C/S: A Robust Two-Factor Authentication and Key Agreement Protocol for C/S Data Transmission in Federated Learning

Chao Huang, Bin Wang *, Zhaoyang Bao and Wenhao Qi

School of Information Electronic Technology, Jiamusi University, Jiamusi 154007, China
* Correspondence: wangbin@jmsu.edu.cn

Abstract: As a hot technology trend, the federated learning (FL) cleverly combines data utilization and privacy protection by processing data locally on the client and only sharing model parameters with the server, embodying an efficient and secure collaborative learning model between clients and aggregated Servers. During the process of uploading parameters in FL models, there is susceptibility to unauthorized access threats, which can result in training data leakage. To ensure data security during transmission, the Authentication and Key Agreement (AKA) protocols are proposed to authenticate legitimate users and safeguard training data. However, existing AKA protocols for client–server (C/S) architecture show security deficiencies, such as lack of user anonymity and susceptibility to password guessing attacks. In this paper, we propose a robust 2FAKA-C/S protocol based on ECC and Hash-chain technology. Our security analysis shows that the proposed protocol ensures the session keys are semantically secure and can effectively resist various attacks. The performance analysis indicates that the proposed protocol achieves a total running time of 62.644 ms and requires only 800 bits of communication overhead, showing superior computational efficiency and lower communication costs compared to existing protocols. In conclusion, the proposed protocol securely protects the training parameters in a federated learning environment and provides a reliable guarantee for data transmission.

Keywords: federated learning (FL); client–server; authentication; hash-chain; elliptic curve cryptography (ECC)

check for updates

Citation: Huang, C.; Wang, B.; Bao, Z.; Qi, W. 2FAKA-C/S: A Robust Two-Factor Authentication and Key Agreement Protocol for C/S Data Transmission in Federated Learning. *Appl. Sci.* **2024**, *14*, 6664. https://doi.org/10.3390/app14156664

Academic Editor: Gianluigi Ferrari

Received: 30 May 2024
Revised: 22 July 2024
Accepted: 25 July 2024
Published: 30 July 2024

1. Introduction

Federated learning (FL) is an innovative approach where multiple organizations can collaboratively develop machine learning models while maintaining data confidentiality. In FL, data is processed locally at each client's site and only model updates, not the raw data, are shared with the server. This method not only enhances privacy but also allows for the pooling of knowledge across different entities, significantly improving model accuracy and utility. For example, in healthcare, FL enables institutions to collaborate on refining disease diagnosis models without exposing individual patient data, ensuring privacy and compliance with regulations [1,2]. Similarly, in consumer technology, companies like Google LLC and Apple Inc. employ FL to improve user experiences in keyboard prediction and voice recognition without compromising user privacy [3].

Despite its potential in enhancing data privacy, federated learning faces significant security challenges in practical applications, as evidenced by multiple data breach incidents in 2023 [4]. These unauthorized accesses involved not only personal information but also crucial federated learning data such as model parameters and gradients during client–server (C/S) data transmission. The compromised data ranged from hundreds of GBs to several TBs, affecting various industries including education, finance, and technology. These unauthorized accesses directly threaten the security of data and the reliability of the learning process, particularly in industries relying on precise data exchanges.

To safeguard data transmission in FL, the AKA protocols [5–9] are proposed to authenticate legitimate users and safeguard training data. However, existing AKA protocols for client–server (C/S) architecture show security deficiencies. These vulnerabilities include inadequate protection for user anonymity and traceability [6,8], potentially allowing attackers to identify and track user activities through data interactions [7]. Furthermore, these protocols [6–8] are susceptible to replay attacks, where attackers resend valid packets to deceive the authentication system, leading to unauthorized data access. Additionally, they are prone to Denial-of-Service (DoS) attacks, which can drain the system by overwhelming it with excessive requests, thereby denying service to legitimate users, among other potential security flaws. Therefore, it is crucial to design a secure and efficient authentication protocol to protect data transmission in FL.

In this paper, we propose a robust 2FAKA-C/S protocol based on ECC and Hash-chain technology that resists attacks from the extended Canetti–Krawczyk (eCK) adversary model [9]. In this enhanced security protocol, ECC leverages the complexity of the discrete logarithm problem to generate a strong yet compact encryption key, reducing computational demands and significantly enhancing the overall security, while also ensuring the forward secrecy. Additionally, the introduction of Hash-chain technology provides further reinforcement to the system's security. By generating unique Hash-chain's hash values for each session, this technique effectively prevents replay attacks, safeguarding the uniqueness and security of every session. The integration of these techniques significantly boosts the security and efficiency of the authentication and key agreement process, rendering the protocol highly suitable for C/S data transmission in FL.

Our contributions are summarized below:

1. 2FAKA-C/S Protocol Design:
   Based on the technology of Hash-chain and ECC, we present a robust and effective 2FAKA-C/S protocol, overcoming the limitations of existing AKA methods by enhancing security while maintaining computational efficiency.
2. Security Analysis:
   In the formal security analysis, we prove the semantic security of the session key using provably secure methods. Through the informal security analysis, we demonstrate that the proposed protocol not only possesses the properties of mutual authentication and forward security but also effectively resists replay attacks, identity impersonation attacks, man-in-the-middle attacks, etc., as well as possesses the ability to protect against other potential network security threats.
3. Performance Analysis:
   Specifically, the proposed protocol is assessed for its performance in a C/S FL environment, with special focus on total runtime and total communication costs. By comparing with existing protocols, 2FAKA-C/S shows significant advantages in these two key performance metrics. Compared with existing solutions, the proposed protocol effectively improves the operational efficiency of the system.

2. Related Work

FL is a collaborative approach that enhances data privacy by allowing multiple organizations to train models without sharing raw data. Ensuring high-quality data contributions is critical for robust model performance. Taïk et al. [10] and Yurochkin et al. [11] focus on improving model accuracy by prioritizing clients with high-quality data and using evaluation metrics. Deng et al. [12] and Shi et al. [13] introduce dynamic participation and continuously monitor the data quality to optimize learning processes. Lastly, Mazzocca et al. [14] described Trustworthy Federated Learning as a Service (TruFLaaS), a trustworthy federated learning service. TruFLaaS emphasized the importance of robust security and trust mechanisms to ensure data privacy and security in the federated learning process.

Considering the security threats posed by unauthorized access to this data, we need authentication mechanisms to ensure the secure transmission of data in the FL scenario. Since the first introduction of a 2FA protocol, a plethora of subsequent protocols have been designed to improve upon the deficiencies of earlier designs. Ding et al. [6] introduced an innovative anonymous authentication protocol tailored for IoT devices using ECC combined with signcryption. This protocol significantly reduces computational and communication overheads, but it fails to fully secure user anonymity, resulting in potential vulnerabilities for user activity tracking and device traceability.

Liu et al. [7] developed a 2FA protocol that uses ECC to enhance security in mobile computing. While it effectively authenticates users and aggregated servers, it remains susceptible to Denial of Service (DoS) attacks and does not adequately address ephemeral session linkability (ESL) attacks within the extended Canetti–Krawczyk (eCK) security model.

In the healthcare scenario, Singh et al. [15] developed a secure authentication protocol using Kyber and AES-GCM encryption amidst the digital health records transition accelerated by the pandemic. This protocol enables secure transactions without physical key exchanges and generates unique session keys. Xu et al. [16] presented a mutual authentication protocol for tele-surgery, utilizing ECC and biometrics to secure data transmission between surgeons and robotic arms. However, both protocols exhibit performance drawbacks, particularly in computational efficiency, which is critical in healthcare applications.

In the domain of wearable and mobile computing, Tu et al. [17] designed an EAKE-WC protocol for wearable computing that tackles privacy and security issues using ASCON [18], XOR operations, and hashing for user and device authentication, ensuring secure communication through unique session keys. Yang et al. [8] proposed an ultra-lightweight authentication method for mobile edge computing, protecting patient biomedical data in IoT-driven WBANs with a two-stage modal square root technique. However, this method, while reducing computational and communication costs, fails to ensure user anonymity and device un-traceability.

Furthermore, Qi et al. [19] tackled the challenge of ensuring item traceability within industrial IoT (IIoT), introducing a robust auditing mechanism capable of withstanding manipulations by malicious cloud servers. This mechanism represents a broader trend towards employing sophisticated auditing techniques to enhance data integrity in IIoT.

Additionally, in the IoT-Edge-Cloud ecosystem, Seifelnasr et al. [20] unveiled Mutual Authentication Privacy-preserving protocol with Forward Secrecy (MAPFS), a mutual authentication protocol that eliminates reliance on cloud administrators for session key establishment and incorporates zero-knowledge proofs to enhance security and autonomy.

3. Preliminaries

This section provides the essential background information that forms the basis for our subsequent discussion.

3.1. System Model

As shown in Figure 1, our system model contains two main entities: the user (or client) (U_i) and the aggregation server (S_j). In Figure 1, the blue lines indicate the registration phase, while the orange lines denote the login and mutual authentication phases.

In the registration phase, U_i sends a registration request to the aggregation server S_j. After receiving the request, the S_j creates a smart card that contains key information, such as the user's identity information, the generated random number, and the aggregation server's public key. Finally, S_j sends the smart card to U_i to enable that U_i can perform subsequent login and mutual authentication operations.

In the following login and mutual authentication phase, U_i and S_j run the operations to obtain mutual authentication and then a session key will be established between entities.

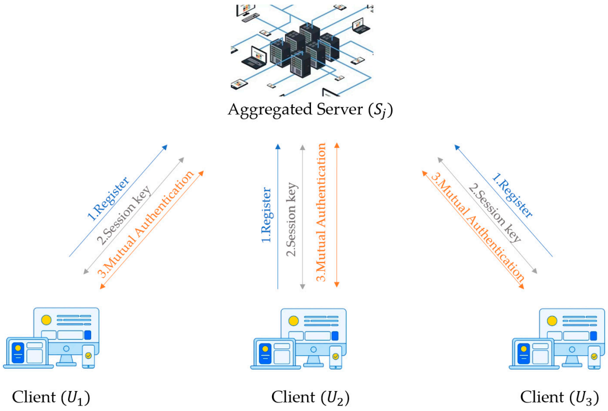

Figure 1. System model of 2FAKA-C/S protocol.

3.2. Elliptic Curve Cryptography (ECC) and Hash-Chain

Elliptic Curve Cryptography (ECC): ECC [21] is a public-key encryption technique based on the elliptic curves' mathematical properties. Compared to traditional cryptographic methods such as RSA, ECC offers the same level of security with much smaller key sizes, leading to increased computational efficiency and reduced storage and bandwidth requirements.

- Elliptic Curve Overview: An elliptic curve is defined by the equation $y^2 = x^3 + ax + b \ mod \ q$, where a and b are parameters and q is a large prime number. The set of points on the curve, along with the point at infinity O, forms a group. For the curve to have a valid group structure, the parameters must satisfy $4a^3 + 27b^3 \not\equiv 0 \ (mod \ q)$.
- Key Generation: In ECC, the private key is a randomly selected integer d. The corresponding public key is obtained by multiplying the base point P on the curve by the private key, resulting in $Q = d{\cdot}P$. This multiplication is effectively repeated addition of the point P d times.
- Group Operations: The addition of two points P and Q on the elliptic curve results in another point $R = P + Q$. Scalar multiplication, denoted as $k{\cdot}P$, involves adding the point P to itself k times, which is fundamental to ECC operations.
- Security Foundation: The security of ECC is based on the elliptic curve discrete logarithm problem (ECDLP). Given two points P and $Q = k{\cdot}P$, it is computationally infeasible to determine the scalar k, especially when q is large. This difficulty ensures the robustness of ECC against various cryptographic attacks.
- Encryption and Decryption: ECC encryption transforms a message into a point on the elliptic curve using the recipient's public key. The sender selects a random integer k and computes $C_1 = k{\cdot}P$ and $C_2 = M + k{\cdot}Q$, where M is the message point and Q is the recipient's public key. The recipient decrypts the message by calculating $M = C_2 - d{\cdot}C_1$ using their private key dd.
- Computational Efficiency: ECC achieves high levels of security with shorter key lengths, resulting in faster computations and lower resource usage. This makes ECC particularly suitable for environments with limited computational power, such as mobile devices and embedded systems.

Hash-chain: The Hash-chain involves the repeated application of a cryptographic hash function to produce a sequence of hash values.

- Construction: A Hash-chain starts with a seed value x, and each subsequent value is generated by hashing the previous value. For example, starting with x, the chain progresses as $h(x), h(h(x)), h(h(h(x)))$, and so on. Originally proposed by Lamport [22], a Hash-chain is a fundamental cryptographic method used to prevent the theft of user passwords over public channels.
- Features: One of the key characteristics of a Hash-chain is its one-way property: it is computationally feasible to compute the chain in the forward direction but infeasible

to reverse the process. This means that given $h^N(x)$, it is practically impossible to determine $h^{N-1}(x)$. However, given $h^{N-1}(x)$, it is straightforward to verify $h^N(x)$. This property makes Hash-chain ideal for resisting replay attacks [23].

3.3. Notations

For clarity and ease of reference, Table 1 shows the notations used in our designed protocol.

Table 1. Notations with related descriptions.

Notation	Description	Notation	Description
U_i	User/Client	x	Long − term key of S_j
S_j	Aggregated Server	$\mathcal{A}$	Malicious adversary
ID_i	Unique identity of U_i	$\|$	String concatenation operation
PID_i	Pseudo − identity of U_i	T	Current timestamp
PW_i	Password chosen by U_i	$\bigoplus$	Bitwise XOR operation
b	Random numbers of S_j	$H(\cdot)$	One-way hash function
a	Random numbers of U_i	SK	Session key shared between U_i and S_j
$\mathcal{D}_{ID}$	Space of user identities.	r_i	Random number chosen by S_j during registration
$\mathcal{D}_{PW}$	Space of user passwords	R_i	Pre − sec ret value chosen by U_i
G	Abelian group on the elliptic curve	SC_i	Smart card issued to U_i
P	Generator of group G	Sum	Counter for allowed login failures
n_0	Random number within the range 2^4 to 2^8	t_u	Timestamp generated by SC_i
PW_i^{new}	New password set by U_i	I	Any instance if no distinction is needed
U^i	i_{th} instance of user U.	S^j	j_{th} instance of server S
q_h	Number of hash queries made by adversary $\mathcal{A}$	q_e	Number of execute queries made by adversary $\mathcal{A}$
q_s	Number of send queries made by adversary $\mathcal{A}$	l	Bit length of the hash value

3.4. Adversary Model

In the adversary models outlined in [24,25], an eCK adversary $\mathcal{A}$ has control over the communication channel between the parties involved, enabling them to perform malicious activities such as intercepting, eavesdropping, and altering transport messages. The adversary's specific capabilities within AKA protocols are detailed below:

1. Through power analysis [26] or other side-channel techniques [27], adversary $\mathcal{A}$ can extract the parameters stored in the user's smart card.
2. $\mathcal{A}$ can intercept, eavesdrop on, and modify transmitted messages in the public channel.
3. $\mathcal{A}$ can list all pairs (PW_i, ID_i) in $(\mathcal{D}_{PW}, \mathcal{D}_{ID})$ in polynomial time, where $\mathcal{D}_{ID}$ and $\mathcal{D}_{PW}$ are the identifier and password spaces.
4. $\mathcal{A}$ can also sign up as a valid user.
5. $\mathcal{A}$ may obtain old session keys (e.g., through digital forensic techniques [24]) due to improper erasure.
6. To evaluate forward secrecy, assume $\mathcal{A}$ has the server's long-term private key.
7. $\mathcal{A}$ can get ephemeral secrets likes random numbers.

4. The Proposed 2FAKA-C/S Protocol

To enhance security for both mobile users and aggregated servers, our proposed protocol integrates key methodologies, which are shown as follows:

1. Utilization of Elliptic Curve Cryptography (ECC): We harness ECC's capability to generate robust cryptographic keys of reduced length, significantly bolstering encryption without increasing computational demands. This is achieved by combining the strengths of ECC with the Diffie–Hellman key exchange protocol. The Diffie–Hellman algorithm allows two users, without a secure communication channel, to generate a shared secret key using random numbers. The Diffie–Hellman of ECC ensures forward secrecy.

2. Implementation of Hash-chain: The proposed protocol incorporates Hash-chain by applying a one-way hash function, H(·), which plays a pivotal role in safeguarding data integrity and strengthening authentication processes. Specifically, Hash-chains generate a unique hash value for each session, ensuring that even if data is intercepted, it cannot be reused in subsequent sessions. This approach is crucial in mitigating risks associated with replay attacks.
3. Mutual Authentication and Session Key Agreement: A pivotal aspect of the proposed protocol is the mutual authentication process, where both the user and the aggregated server authenticate each other's identity before establishing a shared session key for encrypted communications, ensuring both confidentiality and integrity of data.
4. Enhanced Anonymity and DoS Attacks Mitigation: The proposed protocol introduces a pseudo-identity PID_i to preserve user anonymity and prevent traceability. Furthermore, the integration of timestamps T plays a crucial role in defending against DoS attacks by verifying the timeliness of each session.

In the forthcoming sections, this paper gives the protocol's system setup phase, registration phase, subsequent login and authentication phase, and finally, the password update phase for user.

4.1. System Setup Phase

The aggregated server S_j independently chooses a random number $x \in Z_p^*$ and sets an abelian group G in the elliptic curve, where P is a generator of G. Further, S_j selects a one-way hash function $H(\cdot)$. Finally, S_j publicizes the parameter $H(\cdot)$ and reserves a secret long-term key x.

4.2. Registration Phase

To obtain authentication from S_j, the user U_i needs to carry out the following registration steps (R. 1–3) and then complete the operation of registration in the terminal of S_j.

R. 1. *The user U_i chooses an identity ID_i and a pre-secret value R_i; by the secure channel, U_i transmits $\{ID_{\ i}, R_i\}$ to the aggregated server S_j.*

R. 2. *Upon receiving $\{ID_{\ i}, R_i\}$, S_j selects a random number $r_i \in Z_p^*$ and computes $PID_i = H(ID_i \parallel x)$ and $V_i = H(PID_i \parallel x \parallel r_i)$. S_j then stores $\{PID_i, r_i, Sum = 0, R_i\}$ in its database, where Sum indicates the number of allowed login failures before the smart card is revoked. If the number of login failures exceeds the allowed limit, the smart card is marked as revoked in the server's database, and the user is notified to request a new smart card [28,29]. Finally, S_j writes $\{H(x),\ P,\ V_i, PID_i\}$ to a new smart card SC_i and securely transmits SC_i to U_i.*

R. 3. *Upon the user U_i obtaining the smart card SC_i from S_j, SC_i selects $a_i\, \epsilon Z_p^*$ and randomly generates a number $2^4 \leq n_0 \leq 2^8$. Then, SC_i calculates parameters: $RPW_i = H(ID_i \parallel PW_i)\ mod\ n_0, B_i = H(RPW_i \parallel a_i) \bigoplus V_i,\ A_i = H(ID_i \parallel PW_i \parallel a_i)\ mod\ n_0$. Finally, SC_i contains the parameters: $\{a_i, A_i, B_i, X, P, n_0\}$.*

The operations are also summarized in Figure 2, to provide researchers with a quick understanding of the registration phase.

4.3. Login and Mutual Authentication Phase

After U_i registers with S_j effectively, U_i runs the login operation (L.1) and subsequent authentication steps (Auth.1–Auth.2) with S_j.

L. 1. *U_i inputs ID_i', PW_i' to SC_i. Then, SC_i computes value $A_i' = H\left(ID_i' \parallel\ PW_i' \parallel a_i\right)\ mod\ n_0$ and checks whether $A_i' = A_i$. If not, SC_i refuses the login request. Otherwise, SC_i computes $RPW_i' = H\left(ID_i' \parallel\ PW_i'\right)\ mod\ n_0$ and $(V_{\ i} \parallel H(x) \parallel R_i) = B_i \oplus H\left(RPW_i' \parallel a_i\right)$. Subsequently, SC_i picks $a_i \in Z_p^*$ and computes $K_1 = a \cdot P, K_2 = H(x) \oplus R_i$. At this point, SC_i generates a timestamp t_u and* *computes $V_{ii} = H(V_i \parallel t_u)$ and $M_2 = E_{R_i}(PID_i^* \parallel V_i \parallel V_{ii})$, where $E_{R_i}(\cdot)$ is a symmetric encryption algorithm (i.e., AES [30]). Finally, SC_i sends $\{M_2, K_1, K_2, t_u\}$ to S_j.*

Auth. 1. *After obtaining $\{M_2, K_1, K_2, t_u\}$, S_j first verifies the timestamp t_u to ensure that $|T - t_u| < \Delta T$, where T is the current timestamp and ΔT represents the acceptable time gap, thereby ensuring the request falls within the designated time window to prevent DoS attacks. S_j calculates $R_i = H(x) \oplus K_2, PID_i^* \parallel V_i^* \parallel V_{ii}^* = D_{R_i}(M_2)$, where $D_{K_2}(\cdot)$ is a symmetric decryption algorithm. Then, S_j searches PID_i^* in its database. If PID_i^* cannot be found, or the timestamp t_u is outside the acceptable range, or V_{ii}^* does not match $H\left(V_i^* \parallel t_u\right)$, this session is aborted. Otherwise, S_j moves to the next step. S_j extracts r_i^* stored in the database and checks whether $V_i^* = H\left(ID_i^* \parallel x \parallel r_i^*\right)$. If they are unequal, S_j gets that U_i's smart card has been broken. Otherwise, S_j picks $b \in Z_p^*$ and computes $K_3 = b{\cdot}P, K_4 = b{\cdot}K_1$, $M_3 = H\left(K_3 \parallel K_2^* \parallel V_i^* \parallel ID_i^* \parallel K_4\right)$, $SK_s = H\left(K_4 \parallel ID_i^* \parallel V_i^*\right)$, $R_i^{new} = H(R_i)$ (where R_i^{new} is from the Hash-chain), $PID_i^{new} = H(PID_i \parallel x), C_i = PID_i^{new} \oplus V_i$. S_j updates its database by storing R_i^{new} and PID_i', replacing the previous values of R_i and PID_i, respectively. Lastly, S_j sends the message $\{K_3, M_3, C_i\}$ to U_i openly.*

Auth. 2. *On receiving the message $\{K_3, M_3, PID_i\_V_i\}$, U_i computes $K_4' = a{\cdot}K_3$, $SK_u = H\left(K_4' \parallel ID_i \parallel V_i\right)$, $M_3' = H(K_3 \parallel K_2 \parallel V_i \parallel ID_i \parallel K_4)$ and verifies if $M_3' = M_3$. If the verification fails, the integrity of M_3 may be corrupted and U_i ceases this session. Otherwise, U_i accepts a shared session key $SK = SK_u = SK_s$. Finally, U_i performs the operation $PID_i^{new} = C_i \oplus V_i$ to update the processed identity PID_i. Then U_i computes $R_i^{new} = H(R_i)$ and stores $R_i^{new}\left(\text{i.e.}, B_i = H(RPW_i \parallel a_i) \oplus (V_i \parallel H(x) \parallel R_i^{new})\right)$.*

The operations are summarized in Figure 3 to provide researchers with a quick understanding of the login and authentication phase.

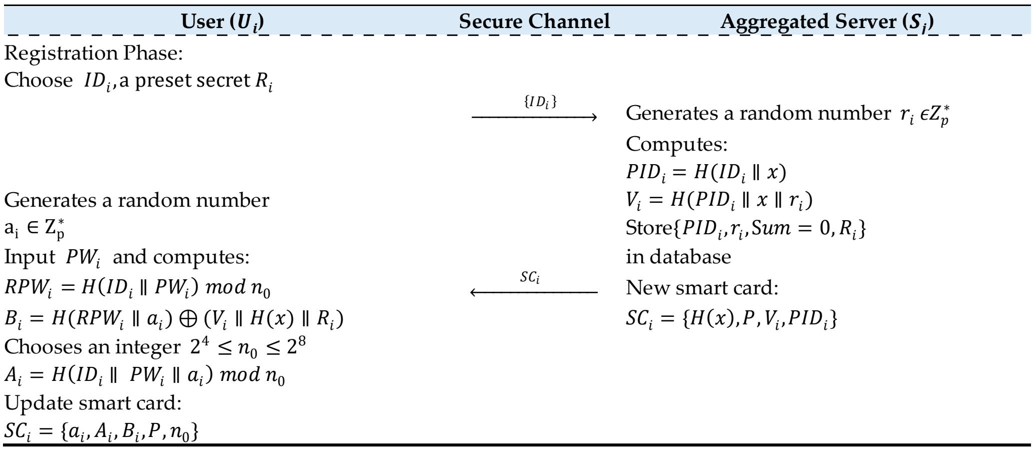
User (U_i)
Secure Channel
Aggregated Server (S_j)
Registration Phase:
Choose ID_i,a preset secret R_i
$\{ID_i\}$
Generates a random number $r_i \epsilon Z_p^*$
Computes:
$PID_i = H(ID_i \parallel x)$
$V_i = H(PID_i \parallel x \parallel r_i)$
Store$\{PID_i, r_i, Sum = 0, R_i\}$
in database
Generates a random number
$a_i \in Z_p^*$
Input PW_i and computes:
$RPW_i = H(ID_i \parallel PW_i) \; mod \; n_0$
$B_i = H(RPW_i \parallel a_i) \oplus (V_i \parallel H(x) \parallel R_i)$
Chooses an integer $2^4 \leq n_0 \leq 2^8$
$A_i = H(ID_i \parallel PW_i \parallel a_i) \; mod \; n_0$
Update smart card:
$SC_i = \{a_i, A_i, B_i, P, n_0\}$
SC_i
New smart card:
$SC_i = \{H(x), P, V_i, PID_i\}$

Figure 2. User registration phase.

4.4. Password Update Phase

In this section, we describe the procedure for users U_i to update their password. Initially, U_i presents their existing or commonly utilized password to the smart card. The smart card then verifies U_i's authenticity by comparing A_i' with A_i and retrieving V_i.

Subsequently, U_i is permitted to set a new password, PW_i^{new}, and accordingly adjust the parameters: $A_i^{new} = H(ID_i \parallel PW_i^{new} \parallel a_i) \; mod \; n_0$, $RPW_i^{new} = H(ID_i \parallel PW_i^{new}) \; mod \; n_0$, and $B_i^{new} = H(RPW_i^{new} \parallel a_i) \oplus V_i$. The process culminates with the smart card updating its stored values from A_i and B_i to A_i^{new} and B_i^{new}, respectively.

User (U_i)	**Public Channel**	**Aggregated Server (S_j)**
Step 1: Input $ID_i^{'}$,$PW_i^{'}$ Compute: $A^{'}_{\ i} = H(ID_i^{'} \parallel PW_i^{'} \parallel a_i)\ mod\ n_0$ Checks if $A_i^{'} = A_i$ Compute: $RPW_i^{'} = H(ID_i^{'} \parallel PW_i^{'})\ mod\ n_0$ $(V_i \parallel H(x) \parallel R_i) = B_i \oplus \mathrm{H}(RPW_i^{'} \parallel a_i)$ Generates a random number a Compute: $K_1 = a \cdot P, K_2 = H(x) \oplus R_i$ $V_{ii} = H(V_i \parallel t_u)$ $M_2 = E_{R_i}(PID_i^* \parallel V_i \parallel V_{ii})$	$\xrightarrow{\{M_1,M_2,K_1\}}$	**Step 2:** Computes: $\|T - t_u\| < \Delta T$ $R_i = H(x) \oplus K_2$ $PID_i^* \parallel V_i^* \parallel V_{ii}^* = D_{R_i}(M_2)$ Checks the validity PID_i^* Check if $V_{ii}^* = H(V_i^* \parallel t_u)$ Extract: r_i^* Check if $V_i^* = H(ID_i^* \parallel x \parallel r_i^*)$ Generates a random number b
Step 4: Computes: $K_4^{'} = a \cdot K_3$ $SK_u = H(K_4^{'} \parallel ID_i \parallel V_i)$ $M_3^{'} = H(K_3 \parallel K_2 \parallel V_i \parallel ID_i \parallel K_4 \parallel R_i^{new})$ Checks if $M_3^{'} = M_3$ $PID_i^{new} = C_i \oplus V_i$ $R_i^{new} = H(R_i)$	$\xleftarrow{\{K_3,M_3\}}$	**Step 3:** Computes: $K_3 = b \cdot P, K_4 = b \cdot K_1$ $M_3 = H(K_3 \parallel K_2^* \parallel V_i^* \parallel ID_i^* \parallel K_4 \parallel R_i^{new})$ $SK_s = H(K_4 \parallel ID_i^* \parallel V_i^*)$ $R_i^{new} = H(R_i)$ $PID_i^{new} = H(PID_i \parallel x)$ $C_i = PID_i^{'} \oplus V_i$

Figure 3. Login and authentication phase.

5. Security Analysis

This section provides a formal security proof and a heuristic analysis to evaluate the security of the proposed protocol, hereafter referred to as $\mathcal{P}$.

5.1. Formal Security Proof

We start with the basic concepts needed for the security proof, then show protocol $\mathcal{P}'s$ security under the elliptic-curve computational Diffie–Hellman (ECCDH) assumption. The ECCDH assumption [31], a variation of the Diffie–Hellman problem, states that given a random pair (aP, bP) in group G, no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ can compute abP with significant advantage.

5.1.1. Basics for Security Proof

The security of $\mathcal{P}$ was assessed based on the BPR2000 [32] and Bresson [33] solutions, also drawing inspiration from the proof techniques of Wang et al. [24]. We detail these fundamentals as follows:

Participants. The proposed $\mathcal{P}$ involves two participants: U and S. Each participant has multiple instances called oracles. The ith instance of U and the jth instance of S are denoted as U^i and S^j, respectively. If no distinction is needed, any instance can be represented as I.

Queries. In a simulated attack scenario, the interaction between participants and adversary $\mathcal{A}$ occurs solely through oracle queries. $\mathcal{A}$ can perform the following types of queries:

- Execute (U^i, S^j): This query simulates the act of eavesdropping on the protocol, capturing all communication records exchanged between U^i and S^j as part of its output.
- Send (I^i, m): This query simulates active attacks by intercepting and blocking a message; the adversary $\mathcal{A}$ sends an imitative message mm. $\mathcal{A}$ then delivers mm to I^i and receives the response from I^i.
- Reveal (I^i): This query discloses the session key to $\mathcal{A}$. If I^i recognizes the session and generates an SK, it returns the session key SK of $I_s^{i\prime}$ to $\mathcal{A}$. Otherwise, it responds with $\perp$, indicating no response.
- Corrupt (U^i): This query allows $\mathcal{A}$ to acquire the secret data held by the user.
- Accepted state: Once an instance I receives and validates the final protocol message, it enters the accepted state. The ordered sequence of the exchanged messages up to

that point is then used to form the session identifier (session ID), for instance I in that particular session.

- Test (I^i): This query tests session key security. A coin b is flipped. If $b = 0$, a random key is sent to $\mathcal{A}$. If $b = 1$, the actual SK is sent. If no SK exists for I^i, $\mathcal{A}$ gets $\perp$. This query can only be used once.

Partnering. Two instances U^i and S^j are partners if: (1) both are in the accepted state or (2) their session identifiers (sid) match, i.e., $sid_U^i = sid_S^j$.

Freshness. An instance I is fresh if: (1) it has computed an accepted session key or (2) no reveal query about SK has been sent to $\mathcal{A}$.

5.1.2. Security Proof

Theorem 1. *Let $Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A})$ denote the advantage of a PPT adversary $\mathcal{A}$ in compromising the semantic security of $\mathcal{P}$ within a limited time t. Given that $\mathcal{A}$ makes q_h hash queries, q_e execute queries, and q_s send queries, we have:*

$$Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A}) \leq 2C' \cdot q_s^{s'} + \Delta$$

where $\Delta = \frac{(q_s+q_h+q_h^2)}{2^{l-1}} + \frac{2(q_s+q_e)^2}{p} + 2q_h Adv_{\mathcal{A}}^{ECCDH}(t')$, where $\mathcal{D}$ represents the password space that follows Zipf's law [34] based on a probability distribution. The parameters s' and C' are related to Zipf's law, l denotes the bit length of the hash value, p is a large prime number, and $t' \leq t + (q_s + q_e + 1)T_c$, where T_c is the time required for an ECC point multiplication operation.

Proof. The detailed proof involves a sequence of games from $Game_0$ to $Game_5$. Let $Pr[E_i]$ represent the probability that $\mathcal{A}$ successfully guesses b in the test-query of $Game_i$, for $i = 0, 1, 2, 3, 4, 5$.

Game E_0. This game simulates a real attack in the context of a random oracle. At the start, a bit b is chosen, thus:

$$Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A}) = |2Pr[E_0] - 1|$$

Game E_1. This game models the random oracle H and a hash list $\Lambda_{\mathcal{H}}$. Furthermore, this game is indistinguishable from the actual execution of the protocol, i.e., game E_0, as all oracles simulate a real attack. Thus, we have:

$$|Pr[E_1] - Pr[E_0]| = 0$$

Game E_2. This game, like game E_1, models all types of queries and terminates under two conditions [35]: (1) a crash from the hash query output and (2) a crash from various records $((M_2, K_1, K_2, t_u), (K_3, M_3))$. According to the birthday paradox, we have:

$$|Pr[E_2] - Pr[E_1]| \leq \frac{q_h^2}{2^{l+1}} + \frac{(q_s + q_e)^2}{2p}$$

Game E_3. This game is modeled similarly to game E_2, but the protocol is aborted if A correctly guesses the authentication parameter M_3 without querying the random oracle. Additionally, this game is difficult to distinguish from game E_2 unless the correct authentication parameter is rejected by U^i or S^j. Hence, we have:

$$|Pr[E_3] - Pr[E_2]| \leq \frac{q_s}{2^l}$$

Game E_4. In this game, the session key SK is obtained without initiating the corresponding random oracle query. Similarly, this game is hard to distinguish from game E_3

unless $\mathcal{A}$ queries the random oracle $\mathcal{H}$ on $(K \| ID_i^* \| V_i^*)$, where $K = ECCDH(K_1, K_3) = abP$ [35]. Therefore, we have:

$$|Pr[E_4] - Pr[E_3]| \leq q_h Adv_{\mathcal{A}}^{ECCDH}(t') + \frac{q_h}{2^l}$$

Game E_5. This game is similar to game E_4, with the additional execution of the test query. If $\mathcal{A}$ initiates a hash $\mathcal{H}$ query with $(abP \| ID_i^* \| V_i^*)$, game E_5 is aborted. Accordingly, SK can be obtained by $\mathcal{A}$ initiating the $\mathcal{H}$ query with a probability of $\frac{q_h^2}{2^{l+1}}$. Alternatively, through a smart-card-loss attack and a corrupt U^i oracle, $\mathcal{A}$ can try to get U^i's password and compromise the session key. Using "fuzzy verifier + honeywords", $\mathcal{A}'s$ chance of guessing the password correctly is no more than $C' \cdot q_s^{s'}$ [24]. To break forward security and obtain the session key, the chance of obtaining abP is at most $\frac{(q_s+q_e)^2}{2p}$. Thus, we have:

$$|Pr[E_5] - Pr[E_4]| \leq C' \cdot q_s^{s'} + \frac{q_h^2}{2^{l+1}} + \frac{(q_s + q_e)^2}{2p}$$

In this game, $\mathcal{A}$ has no advantage in distinguishing the real SK from a session key of the same size created by a random value if $\mathcal{A}$ fails to initiate an $\mathcal{H}$ query with the correct input. Therefore, we have $Pr[E_5] = \frac{1}{2}$.

Finally, according to games $E_0 \sim E_5$ and triangular inequality, we have:

$$\begin{aligned}
Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A}) &= 2\mathrm{Pr}[\mathrm{E}_0] - 1 \\
&= 2(\mathrm{Pr}[\mathrm{E}_0] - \mathrm{Pr}[\mathrm{E}_5]) + 2\mathrm{Pr}[\mathrm{E}_5] - 1 \\
&\leq 2|\mathrm{Pr}[\mathrm{E}_0] - \mathrm{Pr}[\mathrm{E}_5]| \\
&\leq 2|\mathrm{Pr}[\mathrm{E}_0] - \mathrm{Pr}[\mathrm{E}_1] + \mathrm{Pr}[\mathrm{E}_1] - \mathrm{Pr}[\mathrm{E}_2] + \cdots + \mathrm{Pr}[\mathrm{E}_4] - \mathrm{Pr}[\mathrm{E}_5]| \\
&\leq 2|\mathrm{Pr}[\mathrm{E}_0] - \mathrm{Pr}[\mathrm{E}_1]| + 2|\mathrm{Pr}[\mathrm{E}_1] - \mathrm{Pr}[\mathrm{E}_2]| + \cdots + 2|\mathrm{Pr}[\mathrm{E}_4] - \mathrm{Pr}[\mathrm{E}_5]| \\
&\leq 2\mathrm{C}' \cdot \mathrm{q}_\mathrm{s}^{\mathrm{s}'} + \Delta
\end{aligned}$$

where $\Delta = \frac{(q_s + q_h + q_h^2)}{2^{l-1}} + \frac{2(q_s+q_e)^2}{p} + 2q_h Adv_{\mathcal{A}}^{ECCDH}(t')$.

In conclusion, adversary $\mathcal{A}$ has no advantage $Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A})$ in breaking $SK's$ semantic security. □

5.2. Heuristic Analysis

The heuristic analysis can effectively evaluate the protocol's security by its effectiveness, simplicity, and directness [36,37]. This approach enables us to demonstrate that the proposed protocol not only embodies advantageous features (e.g., mutual authentication, forward security, user anonymity) but also exhibits resilience against a spectrum of recognized attacks (e.g., replay attacks, user impersonation attacks).

5.2.1. Mutual Authentication

Within the scope of this protocol, the aggregation server S_j authenticates the user U_i by computing $V_i^* = H(ID_i^* \| \mathrm{x} \| r_i^*)$. In a reciprocal manner, U_i validates the identity of S_j by ensuring the computation result M_3' matches M_3. Upon the completion of this mutual authentication sequence, S_j and U_i will negotiate a shared session key SK. This illustrates that the protocol effectively establishes a robust mutual authentication mechanism between both entities.

5.2.2. Forward Security

The protocol's forward secrecy ensures that all preceding session keys remain intact and secure, even in the event that S_j's long-term key x falls into the hands of an adversary $\mathcal{A}$. When $\mathcal{A}$ intercepts data packets including $\{\{M_2, K_1, K_2\}, \{K_3, M_3\}\}$, he must ascertain $K_4 = b \cdot K_1 = abP$ to deduce the prior session key $SK_s = H(K_4 \| ID_i \| V_i)$. Nonetheless, the formidable computational hurdles in determining the values of a or b, rendering the adversary accurately computing SK. That means the proposed 2FAKA-C/S preserves forward secrecy, safeguarding previous sessions against the compromise of the long-term key.

5.2.3. User Anonymity

In our development of the 2FAKA-C/S protocol, we employ a novel strategy to bolster user anonymity. By allocating a pseudo-identity PID_i to each user and updating it periodically, the proposed protocol significantly enhances both anonymity and untraceability. This pseudo-identity is derived from the user's actual identity $PID_i^{new} = H(PID_i \parallel x)$. This approach effectively obstructs efforts to deduce the user's identity from interaction data. Moreover, we ensure that neither the smart card nor any transmitted data reveal any details that could disclose the user's real or pseudo-identity. Consequently, even if an adversary manages to extract the smart card's parameters $\{a_i, A_i, B_i, P, n_0\}$ or intercept communication data $\{M_2, K_1, K_2, t_u\}$, $\{K_3, M_3\}$, they will be unable to ascertain the user's true or pseudo-identity. Our design not only precludes the direct exposure of user identities but also protects against the tracing of identities and behaviors in the face of persistent communication threats.

5.2.4. Replay Attacks

In replay attacks, $\mathcal{A}$ has the capability to intercept login request data $\{M_2, K_1, K_2\}$ from a prior session and $\mathcal{A}$ attempts to resend this information $\{M_2, K_1, K_2\}$ to the aggregation server S_j; the server will find it impossible to extract any meaningful user identity ID from the replayed data due to the session-by-session refresh of the random parameter R_i. Likewise, if $\mathcal{A}$ tries to resend $\{K_3, M_3\}$ to user U_i, the effectiveness is nullified, as M_3 relies on a session-unique dynamic parameter. Consequently, the adversary will fail to recognize M_3 messages from past sessions in the current context. This mechanism robustly shields against replay attacks.

5.2.5. User Impersonation Attacks

To launch user impersonation attacks based on the exposure of keys, $\mathcal{A}$ requires the precise value of V_i. In the proposed protocol, legitimate users may derive it utilizing the parameter set $\{ID_i, PW_i, B_i, a_i\}$ at their disposal; And the aggregation server might employ the acknowledged parameters r_i and x for an identical computation. However, $\mathcal{A}$ is bound to encounter formidable computational barriers in accessing these key parameters, rendering the emulation of legitimate user $U_i's$ identity towards S_j unfeasible. Consequently, the proposed protocol showcases robust resistance against such impersonation endeavors.

5.2.6. Server Impersonation Attacks

In these attacks, for $\mathcal{A}$ to convincingly mimic the aggregation server S_j, it is imperative to accurately process $\{K_3, M_3\}$. The derivation of M_3 depends on $\left\{R_i, V_i^*, ID_i^*\right\}$, facilitated by a secure hashing algorithm, necessitating $\mathcal{A}'$s comprehensive understanding or viable approximation of these pivotal parameters within a realistic timeframe. Furthermore, $\mathcal{A}$ is required to access key data $\{x, r_i, ID_i^*\}$. The endeavor to decipher these confidential parameters presents a substantial computational impediment to $\mathcal{A}$. Thus, due to the infeasibility of generating precise messages by $\mathcal{A}$, the proposed protocol effectively thwarts impersonation attempts targeting the aggregated server.

5.2.7. Man-in-the-Middle Attacks

Consider a scenario where $\mathcal{A}$ successfully intercepts both the login request $\{M_2, K_1, K_2\}$ and challenge messages $\{K_3, M_3\}$, acquiring all parameters of the smart card SC_i as outlined in the protocol. To launch an effective man-in-the-middle attack, $\mathcal{A}$ must either fabricate new message sequences $\{M_2, K_1, K_2\}$, $\{M_3^*, K_3^*\}$ or replay previously intercepted ones. However, as established, the proposed protocol effectively mitigates replay and impersonation attacks, rendering both the aggregated server S_j and the user U_i incapable of authenticating $\mathcal{A}$. Consequently, the protocol is adeptly fortified against man-in-the-middle attacks.

5.2.8. Temporary Private Key Disclosure Attacks

Within this protocol, secure communication between the user (U_i) and the aggregation server (S_j) is facilitated through designated secret parameters, such as the user's and server's random numbers, a and r_i, respectively. These numbers are pivotal for generating temporary private keys and other authentication parameters throughout the session. Nonetheless, acquisition of these keys by $\mathcal{A}$ does not compromise security due to the insurmountable challenge of producing valid authentication values like V_i. The creation of these values, along with session key negotiation, depends not just on temporary keys but also on a hashed private secret key (x) shared between the user and server. This key remains confidential, thwarting any attempts by adversaries to undermine the authentication mechanism.

5.2.9. Privileged Insider Attacks

Privileged insider attacks denote the misuse of legitimate access by system insiders to exfiltrate training data. During the registration phase, U_i forwards its ID_i to S_j without disclosing any password-related details. S_j then issues an updated smart card SC_i to U_i. Upon receipt, U_i activates SC_i using his exclusively known password PW_i. This procedure secures the new smart card under U_i's possession. Inspection of SC_i's stored parameters reveals the absence of PW_i in plaintext, safeguarding against privileged insider threats effectively.

5.2.10. Denial of Service (DoS) Attacks

In the proposed protocol, an adversary might attempt to disable S_j by replaying old messages. However, S_j initially checks if the time gap meets the condition $|T - t_u| > \Delta T$. If this condition is met, S_j immediately ends the session. Furthermore, should the adversary manipulate the timestamp t_u to satisfy $|T - t_u| < \Delta T$, S_j still dismisses the session due to the subsequent failure in validating the V_{ii}^* value, which is dependent solely on the original timestamp. Thus, such Denial of Service (DoS) attacks are rendered ineffective.

6. Summary Comparison: Functionality and Performance

To highlight the optimal balance between availability and security of our protocol, this section provides a comparative evaluation. We analyze functionalities, communication, and computational overheads, comparing our 2FAKA-C/S protocol with those developed by Ding et al. [6], Liu et al. [7], Yang et al. [8], Roy et al. [32], Hu et al. [38], and Huang et al. [39].

6.1. Security Evaluation Criteria

A critical factor for evaluating the effectiveness of an authentication protocol's functionality is its adherence to foundational principles. Wang et al. [24] and Wang et al. [40] have outlined security criteria specific to AKA (Authentication and Key Agreement) protocols. Building on the security analysis we have provided, Table 2 displays the security criteria established in [24,40], followed by a description of ten evaluation criteria (C*), where C_4 encompasses prevalent attacks such as man-in-the-middle attacks, replay attacks, and de-synchronization attacks, among others.

Table 2. Security evaluation criteria for AKA protocols.

Notation	Description	Description	Description
C_1	Ensure user privacy and anonymity	C_6	Achieve secure key agreement
C_2	Avoid using password verifier tables	C_7	Ensure mutual authentication
C_3	Prevent password leakage	C_8	Operate without clock synchronization
C_4	Defend against known threats	C_9	Have system recovery capabilities
C_5	Protect smart card security	C_{10}	Ensure forward secrecy of communications

6.2. Functionality Comparison

In this section, we present a detailed functional comparison of our protocol against four contemporary protocols [6–8,35], based on the evaluation metrics outlined in Section 5.1.

The outcomes of this comparison are summarized in Table 3, where the notation ✔ means that the protocol demonstrates the property and ✘ denotes that the protocol does not demonstrate the property. According to Table 3, the protocol by Roy et al. does not meet criteria C_5, C_8, and C_{10}. Despite incorporating only three chaotic map operations, the protocol in [35] is unable to ensure forward secrecy when the long-term key is compromised. Moreover, the protocol [35] by Roy et al. stores critical authentication parameters on the smart card and transmits identities in plaintext over public channels, rendering it susceptible to eavesdropping and resulting in potential message desynchronization.

Table 3. Functionality comparison of relevant AKA protocols.

Protocols	Ref.	Evaluation Criteria									
		C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
Roy et al. (2018)	[35]	✔	✔	✔	✔	✘	✔	✔	✘	✔	✘
Yang et al. (2022)	[8]	✘	✔	✔	✔	✔	✔	✔	✔	✔	✔
Ding et al. (2022)	[6]	✘	✔	✔	✘	✔	✔	✔	✔	✔	✔
Liu et al. (2023)	[7]	✘	✔	✔	✘	✔	✔	✔	✔	✔	✔
Hu et al. (2024)	[38]	✔	✔	✔	✘	✔	✔	✔	✔	✔	✔
Huang et al. (2024)	[39]	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
2FAKA-C/S	[-]	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

The protocols outlined in [6–8] do not satisfy criterion C_1, which concerns user anonymity and untraceability. For instance, in the protocol [8], the edge node (EN) transmits the real user identity (ID_{en}) to the access point (AP), thereby directly compromising the user's real identity.

In the case of [6], while the resource device RD does not directly disclose its identity to the IC, the IC is able to ascertain $RD's$ identity via a static index, contradicting the anonymity and untraceability principle that mandates different labels for each authentication attempt.

Similarly, the scheme in [7] compromises user anonymity as it requires user U_i to reveal his real identity ID_i to server S_j in every session, allowing for user tracking. Additionally, in the aspect of C_4, both [7]'s and Hu et al.'s [38] protocols exhibit vulnerability to DoS attacks. Consequently, both protocols also fall short of meeting C_4, further indicating areas for significant improvement in their security apparatus.

Overall, Table 3 indicates that our proposed protocol achieves the outlined security and usability objectives, proving robust against a spectrum of known attacks. While the protocols by Huang et al. [39] also demonstrates strong security and usability features, our protocol remains highly effective.

6.3. Communication and Computation Cost Comparison

To accurately assess and compare the computational and communication overheads, incorporating benchmarks from recent work in [8] for consistent performance evaluation, we define specific terminology for operational timings. Specifically, T_{em}, T_{ea}, T_h, T_{me}, T_{md}, T_{mac}, T_{ae}, T_{ad}, T_{ch}, and T_{fe} represent the execution times for elliptic curve scalar multiplication, elliptic curve point addition, a generic hash function operation, MSR encryption [8], MSR decryption [8], message authentication code computation, AES encryption, AES decryption, Chebyshev map operation, and Fuzzy extractor operation, respectively. Table 4 details these operational runtimes alongside their respective running platforms. Furthermore, to quantify the communication costs incurred during the login and authentication processes, Table 5 outlines the lengths of various security parameters.

Table 4. Summarized runtime of all cryptographic operations (ms).

	T_{em}	T_{ea}	T_h	T_{me}	T_{md}	T_{mac}	T_{ae}	T_{ad}	T_{ch}	T_{fe}
U_i	27.472	0.041	0.006	0.016	0.560	0.118	0.023	0.014	21.02	27.472
S_j	3.823	0.007	0.002	0.003	0.058	0.028	0.010	0.006	21.02	3.823
Terminal	Running platform									
U_i	Raspbian with 1.2-GHz Quad-Core CPU and 1-GB RAM									
S_j	Ubuntu 16.04 LTS with Intel Core i5 7600, 3.5 GHz CPU and 16 GB memory									

Table 5. Length of the safety parameters.

Parameter	Length/Bits
MAC (Message Authentication Code)	160
Timestamp	32
User identity	32
Random number	128
Elliptic curve point	160
The output of the hash function	160
Symmetric ciphertext	128

As depicted in Table 6, the comparative analysis incorporates seven cryptographic protocols [6–8,35,38,39], focusing on their computational and communication costs. The proposed 2FAKA-C/S protocol showcases its competitive edge, with a total running time of 62.644 ms, aligning closely with the fastest protocols like Yang et al. [8] at 62.985 ms and significantly outperforming Huang et al. [39], which has the longest duration at 172.662 ms. This underscores the proposed protocol's efficiency in high-speed processing scenarios.

Table 6. Communication and computation costs in the login and authentication phase.

	Computation Cost			
Protocols	User	Aggregation Server	Total Running Time	Total Communication Cost
Roy et al. (2018) [35]	$9\ T_h + 2T_{ch} + 1T_{fe}$	$5\ T_h + 1T_{ch}$	90.596 ms	960 bits
Yang et al. (2022) [8]	$2\ T_h + 1\ T_{me} + 2\ T_{mac} + 1\ T_{ae}$	$2\ T_h + 1\ T_{md} + 2\ T_{mac} + 1\ T_{ad}$	62.985 ms	608 bits
Ding et al. (2022) [6]	$1T_{em} + 4\ T_h + 1\ T_{mac}$	$5T_{em} + 2\ T_{ea} + 3\ T_h + 1\ T_{mac}$	46.755 ms	832 btits
Liu et al. (2023) [7]	$3T_{em} + 6\ T_h$	$3T_{em} + 4\ T_h$	93.929 ms	768 bits
Hu et al. (2024) [38]	$3T_{em} + 3\ T_h$	$3T_{em} + 3\ T_h$	93.909 ms	864 bits
Huang et al. (2024) [39]	$6T_{em} + 25\ T_h$	$2T_{em} + 17\ T_h$	172.662 ms	3328 bits
2FAKA-C/S protocol	$2T_{em} + 7\ T_h$	$2T_{em} + 6\ T_h$	62.644 ms	800 bits

In terms of communication overhead, the proposed protocol requires only 800 bits, which is more efficient compared to the highest overhead of 3328 bits by Huang et al. [39]. This efficiency is essential in environments where bandwidth is limited. For instance, Roy et al. [35] requires 960 bits, which is significantly higher than our protocol. Similarly, Hu et al. [38] requires 864 bits, which is also higher than our protocol. This comparison underscores the proposed protocol's suitability for bandwidth-constrained environments, making it a particularly effective solution.

In conclusion, the proposed protocol emerges as an eminent solution for secure communication, markedly enhancing processing speed and reducing the communication costs of the proposed protocol.

7. Conclusions

In this study, we have introduced an innovative two-factor authentication protocol (2FAKA-C/S) that can enhance the security of the model parameters transmitted in FL. The comprehensive security analysis confirms its robust defense against a wide array of cyber threats. Furthermore, the protocol's design focuses on minimizing computational demands, making it highly suitable for the resource-constrained environments prevalent in FL. By comparing it with existing protocols, we demonstrated not only its superior security features but also its operational efficiency. Consequently, the proposed protocol is particularly well suited for securing the transmission of model parameters in FL. In our ongoing research, we plan to develop protocols that address inference attacks [41], potentially incorporating techniques like the Gaussian noise addition mechanism as a differential privacy technology to further enhance security.

Author Contributions: Conceptualization, methodology, Writing—Original Draft, C.H.; Writing—Review and Editing, B.W.; Validation, Z.B. and W.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Project supported by the Fundamental Research Funds for the Provincial Universities of Heilongjiang under grant No. 2023-KYYWF-0639 and the Project supported by the Specialized Research Fund for the Doctoral Program of Jiamusi University under grant No. JMSUBZ2022-12.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Dhade, P.; Shirke, P. Federated Learning for Healthcare: A Comprehensive Review. *Eng. Proc.* **2023**, *59*, 230.
2. Gu, X.; Sabrina, F.; Fan, Z.; Sohail, S. A Review of Privacy Enhancement Methods for Federated Learning in Healthcare Systems. *Int. J. Environ. Res. Public Health* **2023**, *20*, 6539. [CrossRef] [PubMed]
3. Hard, A.; Rao, K.; Mathews, R.; Ramaswamy, S.; Beaufays, F.; Augenstein, S.; Eichner, H.; Kiddon, C.; Ramage, D. Federated Learning for Mobile Keyboard Prediction. *arXiv* **2018**, arXiv:1811.03604.
4. 2023 Data Breach Incident Review. Available online: https://zhuanlan.zhihu.com/p/673692169 (accessed on 22 December 2023).
5. Chow, M.C.; Ma, M. A Secure Blockchain-Based Authentication and Key Agreement Scheme for 3GPP 5G Networks. *Sensors* **2022**, *22*, 4525. [CrossRef] [PubMed]
6. Ding, X.; Wang, X.; Xie, Y.; Li, F. A lightweight anonymous authentication protocol for resource-constrained devices in internet of things. *IEEE Internet Things J.* **2022**, *9*, 1818–1829. [CrossRef]
7. Liu, K.; Zhou, Z.; Cao, Q.; Xu, G.; Wang, C.; Gao, Y.; Zeng, W.; Xu, G. A robust and effective two-factor authentication (2FA) protocol based on ECC for mobile computing. *Appl. Sci.* **2023**, *13*, 4425. [CrossRef]
8. Yang, X.; Yi, X.; Khalil, I.; Luo, J.; Bertino, E.; Nepal, S.; Huang, X. Secure and lightweight authentication for mobile-edge computing-enabled WBANs. *IEEE Internet Things J.* **2022**, *7*, 12563–12572. [CrossRef]
9. Liu, X.; Gao, K.; Xue, L.; Chang, G.; Zhou, F. Authenticated Key Exchange in eCK Model. *Comput. Sci.* **2014**, *41*, 172–177.

10. Taïk, A.; Moudoud, H.; Cherkaoui, S. Data-Quality Based Scheduling for Federated Edge Learning. In Proceedings of the 2021 IEEE 46th Conference on Local Computer Networks (LCN), Edmonton, AB, Canada, 4–7 October 2021; pp. 17–23.
11. Yurochkin, M.; Agarwal, M.; Ghosh, S.; Greenewald, K.; Hoang, N.; Khazaeni, Y. Bayesian Nonparametric Federated Learning of Neural Networks. In Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019; pp. 7252–7261.
12. Deng, Y.; Lyu, F.; Ren, J.; Wu, H.; Zhou, Y.; Zhang, Y.; Shen, X. AUCTION: Automated and Quality-Aware Client Selection Framework for Efficient Federated Learning. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 1996–2009. [CrossRef]
13. Shi, Y.; Liu, Z.; Shi, Z.; Yu, H. Fairness-Aware Client Selection for Federated Learning. In Proceedings of the 2023 IEEE International Conference on Multimedia and Expo (ICME), Brisbane, Australia, 10–14 July 2023; pp. 324–329.
14. Mazzocca, C.; Romandini, N.; Mendula, M.; Montanari, R.; Bellavista, P. TruFLaaS: Trustworthy federated learning as a service. *IEEE Internet Things J.* **2023**, *10*, 21266–21281. [CrossRef]
15. Singh, B.M.; Natarajan, J. A novel secure authentication protocol for eHealth records in cloud with a new key generation method and minimized key exchange. *J. King Saud Univ.* **2023**, *35*, 101629.
16. Yang, X.; Yi, X.; Nepal, S.; Khalil, I.; Huang, X.; Shen, J. Efficient and anonymous authentication for healthcare service with cloud based WBANs. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2728–2741. [CrossRef]
17. Tu, S.; Badshah, A.; Alasmary, H.; Waqas, M. EAKE-WC: Efficient and anonymous authenticated key exchange scheme for wearable computing. *IEEE Trans. Mob. Comput.* **2024**, *23*, 4752–4763. [CrossRef]
18. Ascon—Authenticated Encryption & Hashing. Available online: https://ascon.iaik.tugraz.at (accessed on 14 November 2022).
19. Qi, S.; Wei, W.; Cheng, J.; Zheng, Y.; Su, Z.; Zhang, J.; Qi, Y. Secure and efficient item traceability for cloud-aided IIoT. *ACM Trans Sens. Netw.* **2022**, *18*, 171–178. [CrossRef]
20. Seifelnasr, M.; AlTawy, R.; Youssef, A.; Ghadafi, E. Privacy-preserving mutual authentication protocol with forward secrecy for IoT-edge-cloud. *IEEE Internet Things J.* **2024**, *11*, 8105–8117. [CrossRef]
21. Roy, S.; Khatwani, C. Cryptanalysis and Improvement of ECC Based Authentication and Key Exchanging Protocols. *Cryptography* **2017**, *1*, 9. [CrossRef]
22. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]
23. Hakeem, S.A.A.; El-Kader, S.M.A.; Kim, H. A Key Management Protocol Based on the Hash Chain Key Generation for Securing LoRaWAN Networks. *Sensors* **2021**, *21*, 5838. [CrossRef]
24. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans. Dependable Secur. Comput.* **2016**, *15*, 708–722. [CrossRef]
25. He, D.; Wang, D. Robust biometrics-based authentication scheme for multiserver environment. *IEEE Syst. J.* **2015**, *9*, 816–823. [CrossRef]
26. Gattu, N.; Khan, M.; De, A.; Ghosh, S. Power Side Channel Attack Analysis and Detection. In Proceedings of the 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD), San Diego, CA, USA, 2–5 November 2020; pp. 1–7.
27. Stjepan, P.; Guilherme, P.; Luca, M.; Lichao, W.; Lejla, B. SoK: Deep Learning-based Physical Side-channel Analysis. *ACM Comput. Surv.* **2023**, *55*, 1–35.
28. Moon, J.; Lee, D.; Lee, Y.; Won, D. Improving Biometric-Based Authentication Schemes with Smart Card Revocation/Reissue for Wireless Sensor Networks. *Sensors* **2017**, *17*, 940. [CrossRef] [PubMed]
29. Xie, Q.; Liu, W.; Wang, S.; Hu, B.; Dong, N.; Yu, X. Robust password and smart card based authentication scheme with smart card revocation. *J. Shanghai Jiaotong Univ. Sci.* **2014**, *19*, 418–424. [CrossRef]
30. Liu, S.; Li, Y.; Jin, Z. Research on Enhanced AES Algorithm Based on Key Operations. In Proceedings of the 2023 IEEE 5th International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Dali, China, 11–13 October 2023; pp. 318–322.
31. Wang, D.; Wang, N.; Wang, P.; Qing, S. Preserving privacy for free: Efficient and provably secure two-factor authentication scheme with user anonymity. *Inf. Sci.* **2015**, *321*, 162–178. [CrossRef]
32. Bellare, M.; Pointcheval, D.; Rogaway, P. Authenticated key exchange secure against dictionary attacks. In Proceedings of the Eurocrypt 2000, Bruges, Belgium, 14–18 May 2000; pp. 139–155.
33. Bresson, E.; Chevassut, O.; Pointcheval, D. Security proofs for an efficient password-based key exchange. In Proceedings of the 10th ACM Conference on Computer and Communications Security, Washington, DC, USA, 27–30 October 2003; pp. 241–250.
34. Wang, D.; Wang, P. On the implications of Zipf's law in passwords. In *Computer Security—ESORICS 2016, Proceedings of the 21st European Symposium on Research in Computer Security, Heraklion, Greece, 26–30 September 2016*; Askoxylakis, I., Ioannidis, S., Katsikas, S., Meadows, C., Eds.; Springer: Cham, Switzerland, 2016; pp. 111–131.
35. Roy, S.; Chatterjee, S.; Das, A.K.; Chattopadhyay, S.; Kumari, S.; Jo, M. Chaotic map-based anonymous user authentication scheme with user biometrics and fuzzy extractor for crowdsourcing Internet of Things. *IEEE Internet Things J.* **2017**, *5*, 2884–2895. [CrossRef]
36. Zou, S.; Cao, Q.; Wang, C.; Huang, Z.; Xu, G. A robust two-factor user authentication scheme-based ECC for smart home in IoT. *IEEE Syst. J.* **2022**, *16*, 4938–4949. [CrossRef]
37. Bouraghi, H.; Rezayi, S.; Amirazodi, S.; Nabovati, E.; Saeedi, S. Evaluating the usability of a national health information system with heuristic method. *J. Educ. Health Promot.* **2023**, *11*, 182.

38. Hu, S.; Jiang, S.; Miao, Q.; Yang, F.; Zhou, W.; Duan, P. Provably Secure ECC-Based Anonymous Authentication and Key Agreement for IoT. *Appl. Sci.* **2024**, *14*, 3187. [CrossRef]
39. Huang, W. ECC-based three-factor authentication and key agreement scheme for wireless sensor networks. *Sci. Rep.* **2024**, *14*, 1787. [CrossRef]
40. Wang, D.; Gu, Q.; Cheng, H.; Wang, P. The request for better measurement: A comparative evaluation of two-factor authentication schemes. In Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, ASIA CCS '16, Xi'an, China, 30 May–3 June 2016; pp. 475–486.
41. Mehnaz, S.; Dibbo, S.V.; De Viti, R.; Kabir, E.; Brandenburg, B.B.; Mangard, S.; Schneider, T. Are your sensitive attributes private? novel model inversion attribute inference attacks on classification models. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 4579–4596.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.