applied sciences

MDPI

Article

A Robust and Effective Two-Factor Authentication (2FA) Protocol Based on ECC for Mobile Computing

Kaijun Liu [1], Zhou Zhou [2], Qiang Cao [1,*], Guosheng Xu [1], Chenyu Wang [1], Yuan Gao [1], Weikai Zeng [1] and Guoai Xu [1,3]

[1] Key Laboratory of Trustworthy Distributed Computing and Service (MoE), Beijing University of Posts and Telecommunications, Beijing 100876, China
[2] RIOH High Science and Technology Group, Beijing 100088, China
[3] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China
* Correspondence: scq@bupt.edu.cn

Abstract: The rapid development of mobile computing (e.g., mobile health, mobile payments, and smart homes) has brought great convenience to our lives. It is well-known that the security and privacy of user information from these applications and services is critical. Without the prevention provided by an authentication mechanism, safety vulnerabilities may accumulate, such as illegal intrusion access resulting in data leakage and fraudulent abuse. Luckily, the two-factor authentication (2FA) protocols can secure access and communication for mobile computing. As we understand it, existing 2FA authentication protocols weaken security in the pursuit of high efficiency. How efficiency can be achieved while preserving the protocol's security remains a challenge. In this study, we designed a robust and effective 2FA protocol based on elliptic curve cryptography (ECC) for authentication of users and service providers. We proved the robustness (respectively, the effectiveness) of the presented protocol with the heuristic analysis and security verification provided by the ProVerif tool (respectively, with a performance comparison based on six schemes). Performance comparisons in terms of message rounds, communication, and computation overheads showed that our scheme was superior to the exiting schemes or comparable as a whole; i.e., only two rounds, 1376 bits, and 1.818 ms were required in our scheme, respectively. The evaluation results showed that the proposed 2FA protocol provides a better balance between security and availability compared to state-of-the-art protocols.

Keywords: security and privacy; two-factor authentication (2FA); elliptic curve cryptography (ECC); mobile computing; ProVerif

Citation: 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. https://doi.org/10.3390/app13074425

Academic Editor: David Megías

Received: 11 February 2023
Revised: 9 March 2023
Accepted: 14 March 2023
Published: 30 March 2023

1. Introduction

With the rapid development of mobile application services using mobile computing, a variety of mobile applications (e.g., e-mail, social networks, online shopping, playing videos, and mobile games) are becoming more and more practical, which not only enhances people's ways of life but also brings them more convenience [1]. It is worth mentioning that data security and privacy in these services are vulnerable to various threats. The Check Point researchers declared in an analysis report that about 100 million users' private data were leaked due to illegal intrusion from multiple Android applications, which included real-time databases, push notifications, and cloud key storage, and these leaked data may become "fat meat" in the eyes of malicious actors [2,3]. Attention must be given to security and privacy issues as soon as possible.

The two-factor authentication mechanism (i.e., password + smart card) can achieve user identity verification and session key agreement through protocol interaction. Legal users can access data securely via the session key, thus effectively protecting data security and privacy. However, existing two-factor authentication (2FA) protocols have a fly in the

ointment. Given that mobile devices may have constrained resources, many 2FA protocols sacrifice security for higher efficiency and availability.

From the perspective of security, the issue is that the network communication entities in 2FA are subject to diverse attacks, such as impersonation attacks and privileged insider attacks. The schemes in [4,5] do not apply advanced technical means, such as multi-factor authentication and the technology of custom dictionaries, and cannot resist key-compromised user impersonation attacks and password-guessing attacks [6–8].

From the perspective of efficiency, to enhance computational efficiency and diminish the communication overhead, earlier researchers tried to design a practical authentication and key agreement (AKA) protocol by using the hash function and symmetric cryptography (e.g., [9–11]). Gope et al. [9] put forward a lightweight privacy-preserving authentication protocol in which the server does not need to carry out any time-consuming search operations to identify the tag. In addition, it does not need to store a secret key in the tag device. Yang et al. [10] proposed an efficient, perfect forward secrecy-enabled AKA protocol on the basis of a lightweight hash function and XOR operation. Das et al. [11] developed a remote user authentication protocol based on dynamic ID that allows the user to select and update their passwords randomly and does not maintain a verifier table. Nevertheless, it was found that the scheme in [12] could not provide forward secrecy to secure the session key.

Public-key cryptography technology can be used to enhance the security of the AKA protocol [12]. These public-key cryptography technologies (such as ECC [13], RSA [14], and bilinear pairings [15]) are becoming widely used in the design of AKA protocols [7,16,17], making it possible to enhance the safety of the session key and preserve user's anonymity, etc. However, given the authentication performance, using a large number of public-key cryptography techniques throughout the process often leads to greater communication/storage consumption costs and lacks practicality. Accordingly, designing a 2FA protocol that balances security and availability is a challenge.

1.1. Related Work

Since the first 2FA protocol [18] was presented in 1981, hundreds of research studies on 2FA protocols for mobile computing have been undertaken, such as on client–server (C/S) architecture [16,17,19,20] and multi-server environments [21].

On the one hand, for the design of the technical protocol, Durlanik et al. [16] proposed a 2FA protocol implementing a public key exchange mechanism with ECC for the session initiation protocol (SIP). They stated that the memory requirements and total execution times of the proposed protocol were greatly improved compared to non-elliptic approaches. For multi-server environments, Chatterjee et al. [21] introduced a modified authentication protocol employing symmetric key encryption–decryption, the hash function, and a Chebyshev chaotic map and proved that the user can only use a single identity and password to manage authentication for different servers.

On the other hand, to enhance the security of the 2FA protocol, Wang [22] provided a design philosophy, a corresponding solution, and a stronger attack model for 2FA protocols. Later, Wang et al. [19] investigated the difficulty of designing identity-based privacy protection 2FA protocols. To enable trusted users (such as doctors or clinicians) to access sensor data from patients using wireless body area networks in the healthcare IoT, Fotouhi et al. [20] designed a lightweight 2FA protocol. Additionally, security proof results showed that the presented protocol offered forward secrecy and could resist common attacks, including privileged insider attacks.

Furthermore, on the basis of cloud services, Vivekanandan et al. [23] proposed a three-factor mobile user authentication protocol for distributed multimedia in 2020. They stated that their protocol provides extra characteristics, such as user choice-based service provider registration, initial user identity registration, and user revocation. Although the protocol [23] can resist various known attacks, it has low authentication efficiency due to the high computational overhead.

To effectively achieve remote communication between users of specific medical services and service providers, Hsu et al. [24] designed a three-factor, user-controlled, single-sign-on scheme with privacy protection and fast authentication. The results of the performance comparison indicated that their scheme had more security attributes and the lowest cost. However, unlike other schemes that store credentials on the server side, this scheme stores large quantities of user credentials on the client side, which results in low communication efficiency.

For end-to-end communication in 5G-enabled narrow-band IoT networks, Hsu et al. [25] proposed a privacy-preserving authenticated key exchange protocol for a multi-server architecture that allows mobile users to log on to multiple servers with an easy-to-remember password and then compute a session key. Although they used elliptic curve cryptography with a small key size to improve communication efficiency, the protocol bears the risk of the session key being easily obtained by adversaries.

In 2021, in order to resist offline dictionary guessing attacks and continuous leakage of secrets from identity servers, Zhang et al. [26] put forward a password-based threshold single-sign-on authentication protocol for mobile users. In addition, they designed a hybrid mechanism and mixed it with the proposed protocol to effectively thwart online dictionary guessing attacks. However, their solutions are not satisfactory in terms of performance and are not suitable for large-scale applications.

For distributed mobile cloud environments, Vivekanandan et al. [27] put forward a privacy protection user authentication protocol using blockchain technology. By means of security analysis methods (e.g., BAN logic, informal analysis, the scyther tool, and the Automated Validation of Internet Security Protocols and Applications (AVISPA) tool), they examined the proposed protocol in relation to various common attacks and found that the proposed protocol could resist all known attacks. Similarly to [26], Sastry and Reddy's scheme also has the problem that its low performance is not conducive to enhancing the authentication phase efficiently.

To preserve user privacy in the IoT healthcare system, Lin et al. [28] designed a smart card-based authentication protocol with a multi-server architecture. However, we found that no timestamp was used, and so the proposed protocol could not resist denial of service (DoS) attacks. On the basis of extended chaotic maps, Meshram et al. [29] presented a 2FA protocol where a new value SB_i is stored in the server during the authentication procedure. However, this protocol cannot resist desynchronization attacks. Despite the fact that user anonymity can be ensured with the protocols from [28,29], they cannot offer user un-traceability, since the messages in the proposed protocol contain various continual values with which attackers can guess the identities of users easily. Additionally, as the users' private credentials are stored directly in the card without being shielded, Lin et al.'s scheme [28] is vulnerable to stolen smart card attacks.

In 2022, in order to lessen the operation costs and hardware overhead caused by card readers, Meher and Amin [30] designed a multi-factor authentication protocol that does not use smart cards and which is user-friendly and robust. Moreover, the proposed protocol addressed the problem of smart card loss/theft. The authors analyzed their authentication protocol in its response to several security threats, and the results showed that their scheme was safe.

For the multiple service providers in a 6G-assisted intelligent medical environment, Le et al. [31] proposed a three-factor (i.e., smart card, password, and biometrics) authentication protocol with time-limited characteristics. In their scheme, service providers and patients can establish healthcare communications effectively and securely. However, user credentials are stored on the server side, which has potential risks. Moreover, the protocol is vulnerable to password-guessing attacks.

Considering the security threats from physical attacks, physically unclonable functions (PUFs) that can resist physical attacks are widely used to design robust authentication protocols [32,33]. These two research works both claimed that the proposed schemes could resist physical attacks, such as cloning attacks and physical tampering attacks.

1.2. Motivations and Contribution

In practical terms, a 2FA protocol (password + smart card as the two authentication factors) is supposed to offer comprehensive security and various desired characteristics. Table 1 shows the four essential security goals [34] that a 2FA protocol must meet.

Table 1. Goals with related descriptions.

Goal	Description
Anonymity and un-traceability	Identity protection and user un-traceability
Resistance against password-guessing attacks	The attacker cannot grasp the user's password
Session key security	The attacker cannot compute or steal the session key negotiated between the user and service provider [34]
Resistance against impersonation attacks	Server impersonation attacks and key-compromise user impersonation attacks

However, current state-of-the-art 2FA protocols do not meet at least one of the four presented security goals. For instance, according to the acknowledged criteria and heuristic analysis of this paper, the protocol in [35] cannot offer user un-traceability, the protocols in [36,37] cannot resist password-guessing attacks or provide session-key security, and the protocol in [38] is vulnerable to counterfeiting attacks. Similarly, using a heuristic analysis with detailed attack steps, Shin et al. [39] found that the static key and the client's password in the protocols in [40,41] can be obtained by any attacker. To enhance security and maintain high efficiency, we developed a robust 2FA protocol for mobile computing. The key contributions of this paper are as follows:

1. Design of a 2FA protocol for mobile computing

The "Fuzzy-Verifiers" [42] and "Honeywords" [42] techniques, which can be used to construct a fuzzy password verifier and effectively resist password-guessing attacks, were applied in our protocol. Further, based on the hash function and ECC, we designed a 2FA protocol that supports user registration, mutual authentication, and user password updating.

2. The semantic security of the 2FA protocol

The semantic security of the session key was proved with a security proof in our protocol. Additionally, through heuristic analysis, we demonstrated that the proposed protocol meets the ten security evaluation criteria. Furthermore, our protocol's entity authentication, message confidentiality, and session key security were confirmed using the ProVerif [43] tool.

3. Performance analysis of the 2FA protocol

A comparative analysis of the functionality, communication, and computation cost of the proposed protocol was conducted with six common related protocols; i.e., those of Roy et al. [37] (IEEE IoTJ'18), Islam et al. [5] (IEEE IoTJ'18), and so on. A better balance between security and availability was achieved in the proposed protocol according to the comparison results.

2. Preliminaries

In this section, some indispensable preliminaries are presented to facilitate an easy understanding of the following sections.

2.1. System Model

As shown in Figure 1, the system model for the 2FA protocol consists of two entities: the user and the service provider. Note that the blue line corresponds to the registration phase and the green line to the mutual authentication phase. In the registration phase, the secret key value and the long-term key are generated by the service provider. When a user

registers with the service provider, the registration request is sent to the service provider by the user, and then the service provider creates a smart card, which is sent to the user to enable them to complete the registration operation.

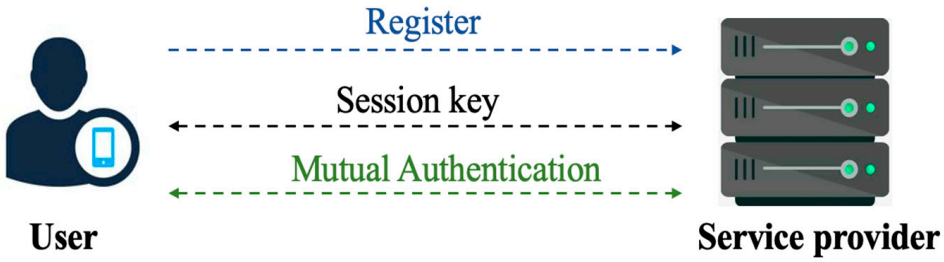

Figure 1. System model of 2FA protocol.

Following the mutual authentication phase, a user with a smart card sends a login request to the service provider, and the service provider then verifies this user according to the received login request. After that, the service provider computes a session key and then conveys the relevant message to the user. Lastly, when receiving the message from the service provider, the user authenticates the identity of the service provider and re-computes the session key.

2.2. Notations

To facilitate understanding among researchers, some notations used in the 2FA protocol are explained in Table 2.

Table 2. Notations with related descriptions.

Notation	Description	Notation	Description
U_i	User	x	Long-term key for S_j
S_j	Service provider	$\mathcal{A}$	Malicious adversary
ID_i	Unique identity of U_i	$\|\|$	String concatenation operation
PW_i	Password chosen by U_i	$\oplus$	Bitwise XOR operation
b	Random numbers for S_j	$H(\cdot)$	One-way hash function
a	Random numbers for U_i	SK	Session key shared between U_i and S_j

2.3. Adversary Model

In the existing adversary models presented in [42,44–52], the communication channel between the communicating parties can be controlled by the adversary, who can initiate malicious operations, such as intercepting, eavesdropping on, and modifying transport messages. In terms of the forward secrecy, $\mathcal{A}$ can also be admitted and corrupt valid parties to obtain long-term keys. In addition, for various reasons (e.g., improper erasure), $\mathcal{A}$ may attain a previous session key. The capabilities of the adversary in 2FA protocols are described below:

1. By means of power analysis or other side-channel techniques, the parameters preserved in the smart card of the user can be obtained by the adversary $\mathcal{A}$;
2. $\mathcal{A}$ can intercept, eavesdrop on, and modify transmitted messages in the public channel;
3. $\mathcal{A}$ can enumerate all pairs (PW_i, ID_i) in $(\mathcal{D}_{PW}, \mathcal{D}_{ID})$ in polynomial time, where $\mathcal{D}_{ID}$ and $\mathcal{D}_{PW}$ represent the spaces of the identifier and password, respectively;
4. $\mathcal{A}$ can also register as a legal user in cases in which anyone can register;
5. $\mathcal{A}$ may be able to obtain previous session keys (e.g., through digital forensic techniques [42]) due to unsuitable erasure;
6. When evaluating the forward secrecy, $\mathcal{A}$ is assumed to have obtained the long-term private key of the service provider.

3. Proposed Protocol

To meet the security requirements for a 2FA protocol for mobile users and service providers, we employed the following five core approaches in the proposed protocol:

1. The user only sends ID_i to the service provider and the protocol uses fuzzy verification technology to design password login verifiers in the registration phase to resist attacks from privileged insiders;
2. To resist password-guessing attacks where the adversary leverages the verifier to guess the password, we used the "Fuzzy-Verifiers" and "Honeywords" technologies [42] to set the verifiers of the password PW_i; i.e., $A_i = H(ID_i \parallel PW_i \parallel a_i) \bmod n_0$, $RPW_i = H(ID_i \parallel PW_i) \bmod n_0$;
3. In terms of guaranteeing efficiency and forward secrecy [53], we applied lightweight ECC to ensure the 2FA protocol's efficiency and, in addition to the long-term key, we added a secret value that cannot be obtained by the adversary in the calculation of the session key to ensure forward secrecy;
4. To resist key-compromise impersonation attacks, we included a secret parameter r_i that can be stored with the service provider securely (e.g., stored in an auxiliary server, as with [17]). Consequently, $\mathcal{A}$ is unable to acquire the value of V_i with r_i to forge the login request message M_1;
5. To ensure the user's un-traceability, a dynamic M_2 computed with the dynamic parameters K_2 and V_i prohibits the adversary from tracing the unchanged identity of the user.

Next, this paper describes the 2FA protocol in detail, including the system setup phase, the registration phase, the following login and authentication phase, and, lastly, the password update phase.

3.1. System Setup Phase

The service provider S_j independently chooses a number $x \, \epsilon \, Z_p^*$, which is a one-way hash function $H(\cdot)$. Then, S_j calculates $X = x \cdot P$ (P is a generator of the abelian group G in the elliptic curve), publicizes the parameter $H(\cdot)$, X, and reserves a long, private, secret key x.

3.2. Registration Phase

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

R. 1 The user U_i chooses an ID_i and, using the secure channel, U_i transmits it to the service provider S_j;

R. 2 Upon receiving $\{ID_i\}$, S_j picks a random number $r_i \epsilon Z_p^*$ and computes $V_i = H(ID_i \parallel x \parallel r_i)$. S_j stores $\{ID_i, r_i, Sum = 0\}$ in its database, where the parameter Sum represents the number of login failures allowed for the user, and the smart card is revoked once the user fails more than Sum times. Finally, S_j adds $\{X, P, V_i\}$ to a fresh smart card SC_i and, using the secure channel, transmits SC_i to U_i;

R. 3 When the user U_i obtains 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 the following parameters: $RPW_i = H(ID_i \parallel PW_i) \bmod n_0, B_i = H(RPW_i \parallel a_i) \bigoplus V_i, A_i = H(ID_i \parallel PW_i \parallel a_i) \bmod n_0$. Finally, SC_i contains the parameters $\{a_i, A_i, B_i, X, P, n_0\}$.

The operations are also summarized in Table 3 to provide researchers with a quick understanding of the registration phase.

Table 3. User registration phase.

User (U_i)	**Secure Channel**	**Service Provider (S_j)**
Registration Phase:		
Choose ID_i	$\{ID_i\}$ ⟶	Generates a random number $r_i \, \epsilon Z_p^*$ Computes: $V_i = H(ID_i \parallel x \parallel r_i)$
Generates a random number $a_i \, \epsilon Z_p^*$ Computes: $RPW_i = H(ID_i \parallel PW_i) \bmod n_0$ $B_i = H(RPW_i \parallel a_i) \bigoplus V_i$ Chooses an integer $2^4 \leq n_0 \leq 2^8$ $A_i = H(ID_i \parallel PW_i \parallel a_i) \bmod n_0$ Update smart card: $SC_i = \{a_i, A_i, B_i, X, P, n_0\}$	⟵ SC_i	Store $\{ID_i, r_i, Sum = 0\}$ in database New smart card: $SC_i = \{X, P, V_i\}$

3.3. 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 (A. 1–A. 2) with S_j:

L. 1 U_i inputs ID_i', PW_i' to SC_i. Then, SC_i computes $A_i' = H\left(ID_i' \parallel PW_i' \parallel a_i\right) \bmod n_0$ and checks whether $A_i' = A_i$. If not, SC_i refuses the login request. Otherwise, SC_i computes $RPW_i' = H\left(ID_i' \parallel PW_i'\right) \bmod n_0, V_i = B_i \bigoplus H\left(RPW_i' \parallel a_i\right)$. Subsequently, SC_i picks $a \epsilon Z_p^*$ and computes $K_1 = a \cdot P, K_2 = a \cdot X, M_1 = H(ID_i \parallel K_1 \parallel K_2 \parallel V_i), M_2 = E_{K_2}(ID_i \parallel V_i)$, where $E_{K_2}(\cdot)$ is a symmetric encryption algorithm. Finally, SC_i sends $\{M_1, M_2, K_1\}$ to S_j;

A. 1 After obtaining $\{M_1, M_2, K_1\}$, S_j calculates $K_2^* = x \cdot K_1, ID_i^* \parallel V_i^* = D_{K_2^*}(M_2)$, where $D_{K_2}(\cdot)$ is a symmetric decryption algorithm. Then, S_j searches ID_i in its database. If ID_i^* cannot be found, 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 understands that U_i's smart card has been broken. Otherwise, S_j moves to the next step. S_j computes $M_1^* = H\left(ID_i^* \parallel K_1 \parallel K_2^* \parallel V_i^*\right)$ and checks whether $M_1^* = M_1$. S_j will end this session if they are unequal, which means that the integrity of M_1 has been corrupted. Otherwise, S_j picks $b \epsilon Z_p^*$ and computes $K_3 = b \cdot P, K_4 = b \cdot K_1$, $M_3 = H\left(K_3 \parallel K_2^* \parallel V_i^* \parallel ID_i^* \parallel K_4\right), SK_s = H\left(K_4 \parallel ID_i^* \parallel V_i^*\right)$. Lastly, S_j sends the message $\{K_3, M_3\}$ to U_i openly;

A. 2 On receiving the message $\{K_3, M_3\}$, U_i computes $K_4' = a \cdot K_3, SK_u = H(K_4' \parallel ID_i \parallel V_i)$, $M_3' = H(K_3 \parallel K_2 \parallel V_i \parallel ID_i \parallel K_4)$ 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 thinks about a shared session key $SK = SK_u = SK_s$.

Again, the operations are summarized in Table 4 to provide researchers with a quick understanding of the login and authentication phase.

3.4. Password Update Phase

Here, the user U_i can change the password; that is, U_i only submits his/her old or frequently used password to the smart card as shown in the login phase. After the smart card recognizes U_i's legitimacy by checking if $A_i' = A_i$ and obtains V_i, U_i can choose a new PW_i^{new} and then updates parameters: $A_i^{new} = H(ID_i \parallel PW_i^{new} \parallel a_i) \bmod n_0$, $RPW_i^{new} = H(ID_i \parallel PW_i^{new}) \bmod n_0, B_i^{new} = H(RPW_i^{new} \parallel a_i) \bigoplus V_i$. Lastly, the smart card replaces A_i and B_i with new parameters A_i^{new} and B_i^{new}.

Table 4. Login and authentication phase.

User (U_i)	Public Channel	Server (S_j)
Step 1: Input ID_i', PW_i' Compute: $A_i' = H(ID_i' \parallel PW_i' \parallel a_i) \bmod n_0$ Checks if $A_i' = A_i$ Compute: $RPW_i' = H(ID_i' \parallel PW_i') \bmod n_0$ $V_i = B_i \bigoplus H(RPW_i' \parallel a_i)$ Generates a random number a Compute: $K_1 = a \cdot P, K_2 = a \cdot X$ $M_1 = H(ID_i \parallel K_1 \parallel K_2 \parallel V_i)$ $M_2 = E_{K_2}(ID_i \parallel V_i)$	$\xrightarrow{\{M_1, M_2, K_1\}}$	**Step 2:** Computes $K_2^* = x \cdot K_1$ $ID_i^* \parallel V_i^* = D_{K_2^*}(M_2)$ Checks the validity ID_i^* Extract: r_i^* Check if $V_i^* = H(ID_i^* \parallel x \parallel r_i^*)$ Compute: $M_1^* = H(ID_i^* \parallel K_1 \parallel K_2^* \parallel V_i^*)$ Checks if $M_1 = M_1^*$ Generates a random number b
Step 4: Computes $K_4' = a \cdot K_3$ $SK_u = H(K_4' \parallel ID_i \parallel V_i)$ $M_3' = H(K_3 \parallel K_2 \parallel V_i \parallel ID_i \parallel K_4)$ Checks if $M_3' = M_3$	$\xleftarrow{\{K_3, M_3\}}$	**Step 3:** Computes $K_3 = b \cdot P, K_4 = b \cdot K_1$ $M_3 = H(K_3 \parallel K_2^* \parallel V_i^* \parallel ID_i^* \parallel K_4)$ $SK_s = H(K_4 \parallel ID_i^* \parallel V_i^*)$

4. Security Analysis

In this section, we describe the formal security proof, the heuristic analysis, and the security analysis using the automated verification tool ProVerif employed to assess the security of the protocol. To facilitate the description, the proposed protocol is abbreviated as $\mathcal{P}$.

4.1. Formal Security Proof

In this part, we first provide the basics for the security proof and then prove the security of $\mathcal{P}$ under the following elliptic-curve computational Diffie–Hellman (ECCDH) assumption.

ECCDH: The hardness assumption of the ECCDH problem, as a variant of the Diffie–Hellman power multiplication [53], indicates that, given a random pair (aP, bP) in G, no probabilistic polynomial time (PPT) adversary $\mathcal{A}$ can effectively compute abP with a non-negligible advantage.

4.1.1. Basics for the Security Proof

The security of $\mathcal{P}$ was assessed using the BPR2000 [54] and Bresson [55] basics, and it was further inspired by the proof work published by Wang et al. [42]. The basics are described below.

Participants. A 2FA $\mathcal{P}$ involves two participants: U and S. Each participant has many different instances called oracles. U's ith instance and S's jth instance are denoted as U^i and S^j, respectively. Additionally, any instance can be expressed as I if there are no differences.

Queries. The interaction between participants and the adversary $\mathcal{A}$ only takes place through oracle queries, which simulate the adversary's abilities in a real attack. The kinds of queries that $\mathcal{A}$ can use are as follows:

- Execute (U^i, S^j): This query catches the eavesdropping of a protocol and, correspondingly, all communication records between U^i and S^j are included in its output;
- Send $(U^i, Start)$: This query represents the initialization of protocol $\mathcal{P}$;
- Send (I^i, m): This query captures active attacks. More specifically, by intercepting and blocking a message, an imitative message m is created by $\mathcal{A}$. Subsequently, $\mathcal{A}$ conveys m to I^i and then obtains the feedback from I^i;

- Reveal (I^i): This query models the misapplication of the session key. When I^i recognizes the session and creates an SK, it returns I^i's session key SK to $\mathcal{A}$. Otherwise, it responds with $\perp$, which means no response;
- Test (I^i): The session key's semantic security is modeled with this query. A coin b is flipped when the query is received. If $b = 0$, a random secret key of the same size as SK is then sent to $\mathcal{A}$. If $b = 1$, then SK is sent to $\mathcal{A}$. A "$\perp$" is sent to $\mathcal{A}$ if no SK for I^i is created. This query can be invoked momentarily (but only once) during the simulation of the adversary;
- Corrupt (U^i): With this query, the secret data preserved by the user can be acquired by $\mathcal{A}$.

Accepted state: When the last prospective protocol message is accepted, an instance I will enter the accepted state. Significantly, the orderly series connection of all communicated messages forms the session identifier for I for the present session.

Partnering. Two instances U^i and S^j become partners if: (1) U^i and S^j are in the accepted state; (2) the session identifiers (sid) of U^i and S^j are the same—i.e., $sid_U^i = sid_S^j$; (3) S^j's partner identifier (pid) is U^i and vice versa.

Freshness. An instance I is fresh if: (1) an accepted session key has been computed by I; (2) a reveal query is not sent to I by $\mathcal{A}$ or its partner.

4.1.2. Security Proof

In this part, the difference lemma [56] is introduced in Lemma 1 and, with this lemma, the advantage from $\mathcal{A}$ corrupting the session key's semantic security is derived by means of a formal theorem.

Lemma 1. Suppose that $E_1, E_2,$ and F are events defined in a probabilistic distribution and further assume that $E_1 \wedge \neg F \Longleftrightarrow E_2 \wedge \neg F$. Then, $|Pr[E_1] - Pr[E_2]| \leq Pr[F]$ holds, where $Pr[\cdot]$ denotes the probability that the event occurs.

Theorem 1. *Define $Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A})$ as the probability of a PPT adversary $\mathcal{A}$ corrupting the semantic security of $\mathcal{P}$ within a limited time t. When $\mathcal{A}$ delivers q_h hash queries, q_e execute queries, and q_s send queries, we obtain:*

$$Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A}) \leq 2C' \cdot q_s^{s'} + \frac{(q_s + q_h + q_h^2)}{2^{l-1}} + \frac{2(q_s + q_e)^2}{p} + 2q_h Adv_{\mathcal{A}}^{ECCDH}(t')$$

where $\mathcal{D}$ represents the password space that coincides with Zipf's law [44] according to a probability distribution, s' and C' refer to the Zipf's law parameters, l denotes the bit length of the hash value, p represents a large prime parameter, and $t' \leq t + (q_s + q_e + 1)T_c$, where T_c is the calculation time for the point multiplication operation of the ECC.

Proof. Assume that the adversary $\mathcal{A}$ can corrupt the security of $\mathcal{P}$. For such circumstances, we put forward an algorithm $\mathcal{B}$ that is able to solve the ECCDH problem. More precisely, $\mathcal{B}$ responds with abP against the instance (aP, bP) of the ECCDH. The proof consists of a series of games: $E_0, E_1, \ldots, E_5$. Let $Pr[E_i]$ denote the valid output b of $\mathcal{A}$ in E_i, where $(i = 0, 1, 2, 3, 4, 5)$.

Game E_0. This game simulates a real attack. $\mathcal{A}$ has access to all the oracles; that is, we get:

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

Game E_1. This game models the random oracle H by managing $\Lambda_{\mathcal{A}}$ and a hash list $\Lambda_{\mathcal{H}}$. In addition, this game cannot be distinguished from the actual conduction of the protocol—i.e., game E_0—as all oracles are modeled as the real attack. Thus, we have:

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

Game E_2. All types of queries are modelled in this game, as in game E_1, and it is terminated in the following two situations [34]: (1) a crash from the hash query output and (2) a crash from various records—$((M_1, M_2, K_1), (K_3, M_3))$. 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 the game E_2 but the only difference is that the protocol is aborted when $\mathcal{A}$ guesses the authentication parameters M_1 and M_3 accurately without initiating the random oracle query. Further, this game is difficult to differentiate from the previous game E_2 unless the accurate 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. The session key SK is attained without accordingly initiating the random oracle query in this game. Correspondingly, this game is difficult to differentiate from the previous game E_3 unless $\mathcal{A}$ queries from the random oracle $\mathcal{H}$ on $\left(K \parallel ID_i^* \parallel V_i^*\right)$, where $K = ECCDH(K_1, K_3) = abP$ [34]. 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 the previous game E_4, and the only distinction is that the *Test* query is additionally executed. When $\mathcal{A}$ initiates a hash $\mathcal{H}$ query with $\left(abP \parallel ID_i^* \parallel V_i^*\right)$, game E_5 is aborted. Accordingly, on the one hand, SK can be obtained from $\mathcal{A}$ initiating the $\mathcal{H}$ query with the maximum likelihood of $\frac{q_h^2}{2^{l+1}}$. On the other hand, by means of a smart-card-loss attack and by modeling the corrupt $\left(U^i\right)$ oracle, $\mathcal{A}$ may expect to obtain the password for U^i and corrupt the session key. Thanks to the "fuzzy verifier + honeywords" technology, the feasibility of $\mathcal{A}$ correctly guessing a password is not more than $C' \cdot q_s^{s'}$ [42]. Lastly, from the perspective of breaking the forward security to obtain the session key, the probability of obtained abP is $\frac{(q_s+q_e)^2}{2p}$ at most. Therefore, we have:

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

Factually, in this game, $\mathcal{A}$ has no advantage from using the same sized session key created by the random value to discriminate the real SK when $\mathcal{A}$ does not manage to initiate a $\mathcal{H}$ query with the correct input; that is, we have $Pr[E_5] = \frac{1}{2}$.

Finally, according to games $E_0 \sim E_5$ and Lemma 1, we have

$$Adv_{\mathcal{P},\mathcal{D}}^{AKA}(\mathcal{A}) \leq 2C' \cdot q_s^{s'} + \frac{(q_s + q_h + q_h^2)}{2^{l-1}} + \frac{2(q_s + q_e)^2}{p} + 2q_h Adv_{\mathcal{A}}^{ECCDH}(t')$$

□

4.2. Heuristic Analysis

Here, we employed heuristic analysis to evaluate the protocol's security since the heuristic method, with its effective, simple, and direct procedure [7], can show that the proposed protocol not only offers desirable properties but is also resistant to various known attacks.

4.2.1. Timely Password Typo Detection

The proposed protocol decreases the computation and communication overhead in cases of input errors or illegal user-initiated attacks. More precisely, in the login phase, the

smart card SC_i verifies the password's validity by checking whether A_i' ? = A_i after the user inputs $\{ID_i', PW_i'\}$. If $A_i' = A_i$, then the request message is transmitted to the service provider by SC_i. Otherwise, this session is terminated. Therefore, the 2FA provides timely password typo detection.

4.2.2. User Anonymity and Un-Traceability

User anonymity refers to hiding part of the user's information during communication, and un-traceability means that the user's identity cannot be tracked. Practically, in order to obtain the user's identity during the communication session, $\mathcal{A}$ needs to extract all parameters $\{a_i, A_i, B_i, X, P, n_0\}$ stored in SC_i and obtain $\{M_1, M_2, K_1\}, \{K_3, M_3\}$ from U_i and S_j, but no identity information is preserved in the user's smart card or conveyed over the open channel in the proposed protocol. For user traceability, $M_1 = H(ID_i \parallel K_1 \parallel K_2 \parallel V_i)$ and $M_2 = E_{K_2}(ID_i \parallel V_i)$ are variable. The user's real identity ID_i cannot be traced by $\mathcal{A}$. Thus, user anonymity and un-traceability can be achieved.

4.2.3. Privileged Insider Attack

A privilege insider attack refers to insiders using legitimate access to steal confidential information in the system. In the registration phase of our protocol, U_i sends ID_i to S_j without any password-related information. Afterwards, an updated smart card SC_i is transmitted to U_i by S_j. U_i activates SC_i by providing PW_i, which is only known to U_i, when receiving SC_i. Finally, U_i obtains the new SC_i. It can be seen that PW_i is unavailable in plaintext by examining the parameters preserved in SC_i. Thus, the proposed protocol can resist privileged insider attacks.

4.2.4. Key-Compromise User Impersonation Attack

In order to launch a key-compromise user impersonation attack, $\mathcal{A}$ must attain the value of V_i, which can be calculated in two ways: (1) the legal user can compute it with known $\{ID_i, PW_i, B_i, a_i\}$ and (2) the service provider can calculate it because of the known r_i and x. However, computational difficulties arise if $\mathcal{A}$ attempts to acquire these critical parameters; that is, $\mathcal{A}$ cannot impersonate the legitimate user U_i for S_j. Therefore, the proposed protocol is resilient against this attack.

4.2.5. Server Impersonation Attack

In the proposed protocol, $\mathcal{A}$ needs to calculate correct $\{K_3, M_3\}$ to impersonate the service provider S_j. Since $M_3 = H(K_3 \parallel K_2^* \parallel V_i^* \parallel ID_i^* \parallel K_4)$ is computed by $\{V_i^*, ID_i^*\}$ and preserved by a secure hash function, $\mathcal{A}$ has to grasp these critical parameters or estimate the valid values in polynomial time. Next, $\mathcal{A}$ must grasp the secret values $\{x, r_i, ID_i^*\}$. In any case, it is computationally challenging for $\mathcal{A}$ to estimate these private parameters in polynomial time. As a result, the messages cannot be calculated by $\mathcal{A}$ correctly, and the proposed protocol is resistant against server impersonation attacks.

4.2.6. Password-Guessing Attack

In the 2FA protocol, $\mathcal{A}$ can eavesdrop on all information through the open channel and extract all parameters from the smart card of the user. Then, analyses of the password-guessing attack can be conducted from two angles: (i) On the one hand, with an unknown user's identity ID_i, $\mathcal{A}$ guesses $\{ID_i', PW_i'\}$ within the dictionary space. Then, $\mathcal{A}$ will be capable of calculating the relevant parameters $RPW_i' = H(ID_i' \parallel PW_i') \bmod n_0$, and $A_i' = H(ID_i' \parallel PW_i' \parallel a_i) \bmod n_0$. Afterwards, $\mathcal{A}$ verifies whether A_i' ? = A_i. Finally, the above procedures are repeated until the password and identity are guessed by $\mathcal{A}$ accurately. Obviously from the perspective of theory, the relevant ID_i' and PW_i' that meet A_i' ? = A_i can be estimated by $\mathcal{A}$ in polynomial time. Factually, the guessing size for the password and identity space can be represented as $\frac{|\mathcal{D}_{ID}|*|\mathcal{D}_{PW}|}{n_0}$, where $2^4 \leq n_0 \leq 2^8$, and $\mathcal{D}_{PW}$ and $\mathcal{D}_{ID}$ represent the guessing spaces of the password and identity, respectively. Thus, the valid password and identity cannot be guessed by $\mathcal{A}$ effectively because $\frac{|\mathcal{D}_{ID}|*|\mathcal{D}_{PW}|}{n_0} \approx 2^{32}$

is larger than the finite Sum that denotes the time allowed by the smart card for an attacker until login failure, when $|\mathcal{D}_{ID}| = |\mathcal{D}_{PW}| = 10^6$ and $n_0 = 2^8$ [41]. (ii) On the other hand, the identity ID_i of the user is likely to be leaked by $\mathcal{A}$. Nevertheless, $\mathcal{A}$ is unable to estimate the password PW_i correctly since there are still a large number of password candidates $\frac{|\mathcal{D}_{PW}|}{n_0} \approx 2^{12}$ that meet the formula $A_i = H(ID_i \parallel PW_i \parallel a_i) \bmod n_0$. Therefore, no matter whether $\frac{|\mathcal{D}_{ID}|*|\mathcal{D}_{PW}|}{n_0} \approx 2^{32}$ or $\frac{|\mathcal{D}_{PW}|}{n_0} \approx 2^{12}$, the authentic prize cannot be attained by the adversary using a guessing attack.

4.2.7. De-Synchronization Attack

The de-synchronization attack interferes with the parameter updates. Although the initial message stream can be blocked by the attacker, the service provider does not have to change any critical parameters in the database. Therefore, the consistency of the following communications will not be affected by this procedure. Furthermore, even if the attacker can block the second message stream, the consistency of communications between the service provider and user will not be influenced by it either, as the smart card will change its data only if M_3' ? $= M_3$ holds. As a result, the proposed protocol is resilient against de-synchronization attacks.

4.2.8. Replay Attack

Suppose that the login request information $\{M_1, M_2, K_1\}$ for the previous session has been acquired by $\mathcal{A}$ over an open channel. When $\mathcal{A}$ replays $\{M_1, M_2, K_1\}$ to S_j, S_j verifies M_1. Since V_i is updated in every successive session, M_1 is also changed to M_1^{new} each time. Therefore, S_j cannot check the previous M_1 in the present session. In addition, if $\mathcal{A}$ replays $\{K_3, M_3\}$ to U_i, the previous message M_3 cannot be checked by U_i in the present session. Accordingly, the proposed protocol is resistant against replay attacks.

4.2.9. Man-in-the-Middle Attack

Assume that $\mathcal{A}$ manages to block and intercept the login request information $\{M_1, M_2, K_1\}$ and the challenge message $\{K_3, M_3\}$ and extracts all parameters for SC_i in the proposed protocol. To initiate an effective man-in-the-middle attack, $\mathcal{A}$ has to falsify the new message stream $\{\{M_1^*, M_2^*, K_1^*\}, \{M_3^*, K_3^*\}\}$ or replay the previous message stream. As mentioned above, the presented protocol is resilient against replay and impersonation attacks. Neither the service provider nor the user can authenticate $\mathcal{A}$ successfully. Thus, the presented protocol can resist man-in-the-middle attacks.

4.2.10. Mutual Authentication

In the presented protocol, S_j verifies U_i by checking whether $M_1^* = M_1$, while U_i checks S_j by verifying if $M_3' = M_3$. After mutual authentication, a common session key SK is negotiated by S_j and U_i; that is, mutual authentication can be achieved safely with the proposed protocol.

4.2.11. Forward Secrecy of the Session Key

The forward secrecy of the session key indicates that, although the long-term key x of S_j is leaked to $\mathcal{A}$, all previous session keys remain safe. Assume that $\mathcal{A}$ eavesdrops further: $\{\{M_1, M_2, K_1\}, \{K_3, M_3\}\}$. To calculate the prior session key $SK = H(K_4 \parallel ID_i \parallel V_i)$, $\mathcal{A}$ needs to know $ID_i \parallel V_i = D_{K_2}(M_2) = D_{a \cdot X}(M_2)$ and $K_4 = b \cdot K_1 = abP$. Further, computational difficulties arise for $\mathcal{A}$ when they try to obtain the stochastic parameters a or b. Thus, $\mathcal{A}$ is unable to calculate SK. Forward secrecy can be achieved successfully with the presented 2FA protocol.

4.3. Formal Verification Analysis Using ProVerif

ProVerif is the latest popular automated verification tool [43]. By running the process in an infinite message space and session simulation, it can verify whether the authentication protocol can: (1) ensure the confidentiality of a specially defined string, (2) ensure the

authentication of all entities, and (3) prevent an attacker from tracking the secret string (i.e., the session key).

4.3.1. Definition of Parameters in ProVerif

In Figure 2, we provide the definitions of the parameters in ProVerif, where the public channel (ch) and secure channel (sch) are used for communication between the user and service provider. Further, SKusecret is a session key for the user. All functions, as well as related equations, are also illustrated. Two queries (i.e., lines 24 and 25) were run to test whether the session key was secure and whether the user could obtain authentication from the service provider.

Line	Definition of Parameters in ProVerif
1	free ch:channel.(*public channel*)
2	free sch:channel[private].(*secure channel*)
3	(*-session key secrets-*)
4	free SKusecret:bitstring[private].
5	(*-constants-*)
6	const P:bitstring.
7	(*-functions-*)
8	fun h(bitstring):bitstring.(*hash function*)
9	fun xor(bitstring,bitstring):bitstring.(*exclusive-or*)
10	fun rox(bitstring,bitstring):bitstring.(*exclusive-or*)
11	fun con(bitstring,bitstring):bitstring.(*string concatenation*)
12	fun mod(bitstring,bitstring):bitstring.(*mod*)
13	fun mul(bitstring,bitstring):bitstring.(*point multiplication*)
14	(*-reduc and equation-*)
15	equation forall m:bitstring,n:bitstring; xor(xor(m,n),n)=m.
16	equation forall m:bitstring,n:bitstring; rox(n,rox(n,m))=m.
17	equation forall m:bitstring,n:bitstring; mul(mul(P,m),n)=mul(mul(P,n),m).
18	reduc forall m:bitstring,n:bitstring; getl(con(m,n))=m.
19	reduc forall m:bitstring,n:bitstring; getr(con(m,n))=n.
20	(*-encryption-*)
21	fun enc(bitstring,bitstring):bitstring.
22	reduc forall m:bitstring,k:bitstring; dec(enc(m,k),k)=m.
23	(*-queries-*)
24	query attacker(SKusecret).
25	query SK:bitstring;inj-event(UserKey(SK))==>inj-event(ServerKey(SK)).

Figure 2. Definition of parameters in ProVerif.

4.3.2. Code for Process in ProVerif

In Figure 3, lines 26–53 of the code (respectively, lines 55–77 of the code) are dedicated to the registration and authentication of the user (respectively, the registration and authentication of the service provider). Then, through the code process ((!User(Idi,PWi)) | (!GWN(x,X,P))) that is used to run two entities' processes in parallel, we can obtain running results for the program codes (see lines 85–89 in Figure 3; i.e., the verification summary). The result in line 87 indicates that the attacker cannot calculate or track the session key SKusecret, and the last line (i.e., line 88) denotes that the event UserKey(UK) is correctly executed after the event ServerKey(SK) and shows that the user has obtained authentication from the service provider.

Line	Code of Process
26	**Code of User:**
27	let User(IDi:bitstring,PWi:bitstring)=
28	out(sch,IDi);
29	in(sch,(X:bitstring,P:bitstring,Vi:bitstring));
30	new ai:bitstring;
31	let RPWi = h(con(con(IDi,PWi),ai)) in
32	let Bi = xor(Vi,h(con(RPWi,ai))) in
33	new n0:bitstring;
34	let Ai = mod(n0,h(con(con(IDi,PWi),ai))) in
35	!
36	(
37	if Ai = mod(n0,h(con(con(IDi,PWi),ai))) then
38	let RPWi = h(con(con(IDi,PWi),ai)) in
39	let Vi = xor(Bi,h(con(RPWi,ai))) in
40	new a:bitstring;
41	let K1 = mul(P,a) in
42	let K2 = mul(X,a) in
43	let M1 = h(con(con(con(IDi,K1),K2),Vi)) in
44	let M2 = enc(con(IDi,Vi),K2) in
45	out(ch,(M1,M2,K1));
46	in(ch,(K3:bitstring,M3:bitstring));
47	let K4 = mul(K3,a) in
48	let SKh = h(con(con(K4,IDi),Vi))in
49	if M3 = h(con(con(con(con(K3,K2),Vi),IDi),K4)) then
50	let SKu = SKh in
51	out(ch,enc(SKusecret,SKu));
52	0
53	).
54	----------
55	**Code of Service Provider:**
56	let Server(x:bitstring,X:bitstring,P:bitstring) =
57	in(sch,(IDi:bitstring));
58	new ri:bitstring;
59	let Vi = h(con(con(IDi,x),ri)) in
60	out(sch,(X,P,Vi));
61	!
62	(
63	in (ch,(M1:bitstring,M2:bitstring,K1:bitstring));
64	let K2 = mul(K1,x) in
65	let IDiVi = dec(M2,K2) in
66	let IDi = getl(IDiVi) in
67	let Vi = getr(IDiVi) in
68	if Vi = h(con(con(IDi,x),ri)) then
69	if M1 = h(con(con(con(IDi,K1),K2),Vi)) then
70	new b:bitstring;
71	let K3 = mul(P,b) in
72	let K4 = mul(K1,b) in
73	let M3 = h(con(con(con(con(K3,K2),Vi),IDi),K4)) in
74	let SKs = h(con(con(K4,IDi),Vi)) in
75	out(ch,(K3,M3));
76	0
77	).
78	----------
79	**process**
80	new x:bitstring;
81	new IDi:bitstring;(*the user's identity*)
82	new PWi:bitstring;(*the user's password*)
83	let X = mul(P,x) in
84	((!User(IDi,PWi)) | (!Server(x,X,P)))
85	----------
86	**Verification summary:**
87	Query not attacker(SKusecret[]) is true.
88	Query inj-event(UserKey(SK)) ==> inj-event(ServerKey(SK)) is true.
89	----------

Figure 3. Code for process in ProVerif.

5. Summary Comparison: Functionality and Performance

To show the better balance of availability and security in the presented 2FA protocol, this section provides a comparative evaluation focusing on the functionality analyses, communication, and calculation overhead in the schemes developed by Tsai et al. [36], Zhu et al. [38], Liu et al. [35], Roy et al. [37], and Islam et al. [5] and in the presented 2FA protocol.

5.1. Security Evaluation Criteria

The hinge that can be used to assess the goodness of the functionality of an authentication protocol is whether the protocol design conforms to the fundamental principles. Wang et al. [42] and Wang et al. [57] provided conclusions regarding the security criteria in terms of AKA protocols. On the basis of our security analysis demonstrated above, Table 5 presents the safety criteria designed in [42,57] and then ten evaluation criteria (EC_*) are described.

Table 5. Security evaluation criteria for AKA protocols.

Notation	Description	Notation	Description
EC_1	User anonymity and un-traceability	EC_6	Provision of key agreement
EC_2	Password verifier table is unwanted	EC_7	Mutual authentication verification
EC_3	Password exposure is avoidable	EC_8	No clock synchronization
EC_4	Timely typo detection	EC_9	Sound capacity for repair
EC_5	No smart-card-loss attack	EC_{10}	Forward secrecy

5.2. Functionality Comparison

In the following, a comprehensive functionality comparison between the 2FA protocol and the five most advanced protocols [5,35–38] is presented using the evaluation indicators mentioned in Section 5.1.

The comparison results are depicted in Table 6, where the notation $\surd$ means that the protocol demonstrates the property, and $\times$ denotes that the protocol does not demonstrate the property. As shown in Table 6, the protocol from [5] cannot fulfill EC_5 and EC_9, since the smart card in [5] stores an explicit password validation parameter and is not resistant against password-guessing attacks and key-compromise user impersonation attacks. Furthermore, smart card revocation of function is not provided in [5].

Table 6. Functionality comparison of relevant AKA protocols.

Protocols	Ref.	Evaluation Criteria									
		EC_1	EC_2	EC_3	EC_4	EC_5	EC_6	EC_7	EC_8	EC_9	EC_{10}
Tsai et al. (2013)	[36]	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\times$
Zhu et al. (2015)	[38]	$\surd$	$\surd$	$\times$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
Liu et al. (2016)	[35]	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\surd$
Roy et al. (2018)	[37]	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\times$	$\surd$	$\times$
Islam et al. (2018)	[5]	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$
2FA protocol	[-]	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

The protocol in [35] cannot provide user anonymity given the transmission of plaintext identities over an open channel, and clock synchronization attacks cannot be resisted; that is, the protocol in [35] does not demonstrate EC_1, EC_8, and EC_9. Further, Tsai et al.'s protocol [34] does not demonstrate EC_9 and EC_{10}, and Roy et al.'s protocol [37] does not demonstrate EC_5, EC_8, and EC_{10}. Specifically, the protocol in [36] cannot achieve forward secrecy due to the storage of secret key values in the corresponding device [36]. Although only three chaotic-map operations are performed in the protocol in [37], it is still incapable of maintaining forward secrecy once the long-term key is compromised. In addition, the password validation parameter is stored explicitly in the smart card and the plaintext identity is transmitted over the open channel in Roy et al.'s protocol [37], which makes it easy for the attacker to intercept, resulting in the information from the communicator not being synchronized. Similarly, Zhu et al.'s protocol [38] is vulnerable to password-guessing attacks and smart-card-loss attacks due to the smart card storing an explicit password validation parameter. Accordingly, [38] does not demonstrate EC_3 and EC_5.

In general, by observing the Table 6, it can be deduced that the proposed 2FA protocol is the only one that meets the expected security and usability goals and is immune to various known attacks. The proposed 2FA protocol is the only protocol that is resistant against diverse known attacks and can meet the ideal safety and availability goals.

5.3. Communication and Computation Cost Comparison

To provide a fair presentation of the computation and communication cost comparison, drawing on previous research work [58,59], notations with the corresponding running time and running platform are shown in Table 7. For the evaluation of the communication overhead in the login and authentication phase, the length of the safety parameters is defined in Table 8.

Table 7. Notations with related abbreviations.

Notation	Description	Time/ms	Running Platform
T_c	The computing time for the extended chaotic-map operation	0.294	Ubuntu 18.04 with Intel i7-4710HQ, 2.5 GHz CPU and 8 G memory
T_m	The computing time for elliptic curve point multiplication	0.294	
T_s	The computing time for the symmetric cryptography operation	0.021	
T_h	The computing time for a one-way hash operation	0.003	

Table 8. Lengths of the safety parameters.

Parameter	Length/Bits
Timestamp	16
User identity	160
Random number	128
Elliptic curve point	160
The output of the hash function	160
The ciphertext of the symmetric encryption/decryption algorithm	128

As shown in Table 9, the total computation cost for the 2FA protocol is 1.818 ms. Compared with other protocols, the 2FA protocol has slightly higher computing costs (and is closest to the cost of the scheme from [35]), allowing it to obtain higher robustness in terms of safety. The total communication cost of the 2FA protocol is 1376 bits. It can be seen that the communication costs of the relevant protocols are slightly lower than that of the 2FA protocol. It can be seen that only two communication message streams are required in the 2FA protocol, while three message streams are required in the protocols from [5,35,36].

Table 9. Communication and computation costs in the login and authentication phase.

Protocols	Computation Cost			Total Communication Cost	Message Rounds
	User	Service Provider	Total Running Time		
Tsai et al. (2013) [36]	$5T_h + T_m$	$5T_h + 3T_m$	1.206 ms	960 bits	3
Zhu et al. (2015) [38]	$4T_h + 2T_c$	$6T_h + 2T_c$	1.206 ms	736 bits	2
Liu et al. (2016) [35]	$6T_h + 3T_c$	$6T_h + 3T_c$	1.8 ms	1280 bits	3
Roy et al. (2018) [37]	$9T_h + 2T_c$	$6T_h + T_c$	0.927 ms	960 bits	2
Islam et al. (2018) [5]	$7T_h + 2T_m + T_s$	$5T_h + 2T_m + T_s$	1.254 ms	768 bits	3
2FA protocol	$10T_h + 3T_m$	$8T_h + 3T_m$	1.818 ms	1376 bits	2

In conclusion, it can be seen that the proposed 2FA protocol is the most suitable for two-factor authentication and key agreement and balances security and availability in mobile computing.

6. Conclusions

The authentication mechanism has always been an effective way of guaranteeing secure communication in mobile computing. However, existing authentication protocols weaken security in the pursuit of high efficiency. In this study, we designed a robust and effective 2FA protocol and then fully proved the protocol's security and good performance.

The safety analysis results using the ProVerif tool demonstrated that the proposed 2FA protocol was able to achieve semantic security, satisfy all ten evaluation criteria ($EC_1 - EC_{10}$), and provide mutual authentication for and preserve the security of the session key. By comparing the performance of six state-of-the-art protocols and the presented 2FA protocol, we showed that the designed 2FA protocol is more practical. Its design ideas presented in our paper are generic and may be used as guidelines to design AKA protocols. As our ongoing research work, we will focus on a more secure authentication protocol that also considers physical attacks for various situations in mobile computing with IPv6 over low-power wireless personal area networks (6LoWPAN) [60].

Author Contributions: Validation, methodology, writing—original draft, K.L. and Z.Z.; writing—review and editing, Q.C., G.X. (Guosheng Xu), C.W. and G.X. (Guoai Xu); validation, Y.G. and W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under grant no. 62102042 and the National Key Research and Development Program of China under grant no. 2021YFB3101500.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. O'Dea, S. Forecast Number of Mobile Users Worldwide 2020–2025. Available online: https://www.statista.com/statistics/218984/number-of-globalmobile-users-since-2010 (accessed on 2 April 2021).
2. Available online: http://px.tcnet.com.cn/news/industry/2568.html (accessed on 10 February 2023).
3. Available online: http://www.360doc.com/content/20/0901/16/71368091_933437844.shtml (accessed on 2 September 2020).
4. Wazid, M.; Das, A.K.; Kumar, N.; Rodrigues, J.J. Secure three-factor user authentication scheme for renewable-energy-based smart grid environment. *IEEE Trans. Ind. Inform.* **2017**, *13*, 3144–3153. [CrossRef]
5. Islam, S.H.; Vijayakumar, P.; Bhuiyan, M.Z.A.; Amin, R.; Balusamy, B. A provably secure three-factor session initiation protocol for multimedia big data communications. *IEEE Internet Things J.* **2017**, *5*, 3408–3418. [CrossRef]
6. Wang, C.; Wang, D.; Tu, Y.; Xu, G.; Wang, H. Understanding node capture attacks in user authentication schemes for wireless sensor networks. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 507–523. [CrossRef]
7. 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]
8. Wang, Q.; Wang, D. Understanding Failures in Security Proofs of Multi-Factor Authentication for Mobile Devices. *IEEE Trans. Inf. Forensics Secur.* **2022**, *18*, 597–612. [CrossRef]
9. Gope, P.; Lee, J.; Quek, T.Q. Lightweight and practical anonymous authentication protocol for RFID systems using physically unclonable functions. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2831–2843. [CrossRef]
10. Yang, Z.; He, J.; Tian, Y.; Zhou, J. Faster authenticated key agreement with perfect forward secrecy for industrial internet-of-things. *IEEE Trans. Ind. Inform.* **2019**, *16*, 6584–6596. [CrossRef]
11. Das, M.L.; Saxena, A.; Gulati, V.P. A dynamic ID-based remote user authentication scheme. *IEEE Trans. Consum. Electron.* **2004**, *50*, 629–631. [CrossRef]
12. Ma, C.G.; Wang, D.; Zhao, S.D. Security flaws in two improved remote user authentication schemes using smart cards. *Int. J. Commun. Syst.* **2014**, *27*, 2215–2227. [CrossRef]
13. Hankerson, D.; Menezes, A.; Vanstone, S. *Guide to Elliptic Curve Cryptography*; Springer Science & Business Media: New York, NY, USA, 2006.
14. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
15. Zhang, F.; Safavi-Naini, R.; Susilo, W. An efficient signature scheme from bilinear pairings and its applications. In Proceedings of the Public Key Cryptography—PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, 1–4 March 2004; pp. 277–290.
16. Durlanik, A.; Sogukpinar, I. SIP authentication scheme using ECDH. *Proc. Work. Acad. Sci. Eng. Technol.* **2005**, *8*, 350–353.
17. Wang, D.; Cheng, H.; He, D.; Wang, P. On the challenges in designing identity-based privacy-preserving authentication schemes for mobile devices. *IEEE Syst. J.* **2016**, *12*, 916–925. [CrossRef]
18. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]

19. Arkko, J.; Torvinen, V.; Camarillo, G.; Niemi, A.; Haukka, T. Security mechanism agreement for SIP sessions. *Doc. RFC* **2003**, *3329*, 1–24.
20. Fotouhi, M.; Bayat, M.; Das, A.K.; Far, H.A.N.; Pournaghi, S.M.; Doostari, M.A. A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care IoT. *Comput. Netw.* **2020**, *177*, 107333. [CrossRef]
21. Chatterjee, S.; Roy, S.; Das, A.K.; Chattopadhyay, S.; Kumar, N.; Vasilakos, A.V. Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment. *IEEE Trans. Dependable Secur. Comput.* **2016**, *15*, 824–839. [CrossRef]
22. Wang, D.; Wang, P. On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions. *Comput. Netw.* **2014**, *73*, 41–57. [CrossRef]
23. Vivekanandan, M.; Sastry, V.N.; Srinivasulu Reddy, U. Efficient user authentication protocol for distributed multimedia mobile cloud environment. *J. Ambient Intell. Hum. Comput.* **2020**, *11*, 1933–1956. [CrossRef]
24. Hsu, C.L.; Le, T.V.; Hsieh, M.C.; Tsai, K.Y.; Lu, C.F.; Lin, T.W. Three-factor UCSSO scheme with fast authentication and privacy protection for telecare medicine information systems. *IEEE Access* **2020**, *8*, 196553–196566. [CrossRef]
25. Hsu, C.L.; Le, T.V.; Lu, C.F.; Lin, T.W.; Chuang, T.H. A privacy-preserved E2E authenticated key exchange protocol for multi-server architecture in edge computing networks. *IEEE Access* **2020**, *8*, 40791–40808. [CrossRef]
26. Zhang, Y.; Xu, C.; Li, H.; Yang, K.; Cheng, N.; Shen, X. PROTECT: Efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage. *IEEE. Trans. Mob. Comput.* **2020**, *20*, 2297–2312. [CrossRef]
27. Vivekanandan, M.; U, S.R. Blockchain based privacy preserving user authentication protocol for distributed mobile cloud environment. *Peer-to-Peer Netw. Appl.* **2021**, *14*, 1572–1595. [CrossRef]
28. Lin, T.W.; Hsu, C.L.; Le, T.V.; Lu, C.F.; Huang, B.Y. A smartcard-based user-controlled single sign-on for privacy preservation in 5G-IoT telemedicine systems. *Sensors* **2021**, *21*, 2880. [CrossRef] [PubMed]
29. Meshram, C.; Ibrahim, R.W.; Deng, L.; Shende, S.W.; Meshram, S.G.; Barve, S.K. A robust smart card and remote user password-based authentication protocol using extended chaotic maps under smart cities environment. *Soft Comput.* **2021**, *25*, 10037–10051. [CrossRef]
30. Meher, B.K.; Amin, R. A location-based multi-factor authentication scheme for mobile devices. *Int. J. Ad Hoc Ubiquitous Comput.* **2022**, *41*, 181–190. [CrossRef]
31. Le, T.V.; Lu, C.F.; Hsu, C.L.; Do, T.K.; Chou, Y.F.; Wei, W.C. A novel three-factor authentication protocol for multiple service providers in 6G-aided intelligent healthcare systems. *IEEE Access* **2022**, *10*, 28975–28990. [CrossRef]
32. Gope, P.; Sikdar, B. Lightweight and privacy-preserving two-factor authentication scheme for IoT devices. *IEEE Internet Things J.* **2018**, *6*, 580–589. [CrossRef]
33. Kaveh, M.; Mosavi, M.R. A lightweight mutual authentication for smart grid neighborhood area network communications based on physically unclonable function. *IEEE Syst. J.* **2020**, *14*, 4535–4544. [CrossRef]
34. Qiu, S.; Wang, D.; Xu, G.; Kumari, S. Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 1338–1351. [CrossRef]
35. Liu, Y.; Xue, K. An improved secure and efficient password and chaos-based two-party key agreement protocol. *Nonlinear Dyn.* **2016**, *84*, 549–557. [CrossRef]
36. Tsai, J.L.; Lo, N.W.; Wu, T.C. Novel anonymous authentication scheme using smart cards. *IEEE Trans. Ind. Inform.* **2012**, *9*, 2004–2013. [CrossRef]
37. 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]
38. Zhu, H.; Hao, X. A provable authenticated key agreement protocol with privacy protection using smart card based on chaotic maps. *Nonlinear Dyn.* **2015**, *81*, 311–321. [CrossRef]
39. Shin, S.; Kobara, K. Security analysis of password-authenticated key retrieval. *IEEE Trans. Dependable Secur. Comput.* **2015**, *14*, 573–576.
40. *IEEE P1363.2/D11*; Standard Specifications for Password-Based Public-Key Cryptographic Techniques. IEEE P1363 Working Group: New York, NY, USA, 2003.
41. Jablon, D.P. Password authentication using multiple servers. In Proceedings of the Topics in Cryptology—CT-RSA 2001: The Cryptographers' Track at RSA Conference 2001, San Francisco, CA, USA, 8–12 April 2001; pp. 344–360.
42. 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]
43. Blanchet, B.; Smyth, B.; Cheval, V.; Sylvestre, M. Proverif 2.02 pl1: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial; Technical Report. 2020. Available online: https://opam.ocaml.org/packages/proverif/proverif.2.02pl1/ (accessed on 5 September 2020).
44. Wang, D.; Wang, P. On the implications of Zipf's law in passwords. In *Computer Security—ESORICS, 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.
45. Wang, D.; He, D.; Wang, P.; Chu, C.H. Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment. *IEEE Trans. Dependable Secur. Comput.* **2014**, *12*, 428–442. [CrossRef]

46. Eisenbarth, T.; Kasper, T.; Moradi, A.; Paar, C.; Salmasizadeh, M.; Shalmani, M.T.M. On the power of power analysis in the real world: A complete break of the KeeLoq code hopping scheme. In *Advances in Cryptology—CRYPTO, Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2008*; Wagner, D., Ed.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 203–220.
47. Kocher, P.; Jaffe, J.; Jun, B. Differential power analysis. In *Advances in Cryptology—CRYPTO' 99, Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 15–19 August 1999*; Wiener, M., Ed.; Springer: Berlin/Heidelberg, Germany, 1999; pp. 388–397.
48. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining smart-card security under the threat of power analysis attacks. *IEEE Trans. Comput.* **2002**, *51*, 541–552. [CrossRef]
49. Wang, D.; Zhang, Z.; Wang, P.; Yan, J.; Huang, X. Targeted online password guessing: An underestimated threat. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 1242–1254.
50. Wang, D.; Cheng, H.; Wang, P.; Huang, X.; Jian, G. Zipf's law in passwords. *IEEE Trans. Inf. Forensic Secur.* **2017**, *12*, 2776–2791. [CrossRef]
51. Agrawal, S.; Das, M.L.; Lopez, J. Detection of node capture attack in wireless sensor networks. *IEEE Syst. J.* **2018**, *13*, 238–247. [CrossRef]
52. He, D.; Wang, D. Robust biometrics-based authentication scheme for multiserver environment. *IEEE Syst. J.* **2014**, *9*, 816–823. [CrossRef]
53. 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]
54. 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.
55. 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.
56. Shoup, V. Sequences of games: A tool for taming complexity in security proofs. *IACR Cryptol. Eprint Arch.* **2004**, *332*. Available online: https://eprint.iacr.org/2004/332 (accessed on 18 January 2006).
57. 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.
58. Wu, F.; Li, X.; Xu, L.; Vijayakumar, P.; Kumar, N. A novel three-factor authentication protocol for wireless sensor networks with IoT notion. *IEEE Syst. J.* **2020**, *15*, 1120–1129. [CrossRef]
59. Srinivas, J.; Das, A.K.; Wazid, M.; Kumar, N. Anonymous lightweight chaotic map-based authenticated key agreement protocol for industrial Internet of Things. *IEEE Trans. Dependable Secur. Comput.* **2018**, *17*, 1133–1146. [CrossRef]
60. Abbas, G.; Tanveer, M.; Abbas, Z.H.; Waqas, M.; Baker, T.; Al-Jumeily OBE, D. A secure remote user authentication scheme for 6LoWPAN-based Internet of Things. *PLoS ONE* **2021**, *16*, e0258279. [CrossRef]