Anonymous and Practical Multi-factor Authentication for Mobile Devices Using Two-Server Architecture

Haiyan Cao and Yong Xie(✉)

Department of Computer Technology and Application, Qinghai University, Xining, China
mark.y.xie@qq.com

Abstract. At present, password authentication technology using single-server architecture has been widely used in practice. However, it cannot resist internal privilege attack, dictionary guessing attack, and other attacks. To solve the above problems, this paper proposes an anonymous and practical multi-factor authentication protocol for mobile devices using two-server architecture with honeywords. The protocol is more secure than the existing single-factor authentication protocols, and can solve serious security problems such as internal privilege attack and direct leakage of private data after the server is compromised using a single server architecture. The strict security analysis proves that the protocol is secure. Compared to similar protocols, our protocol needs lower computation and communication costs, and can better meet the practical application requirements.

Keywords: Mobile devices · Anonymous · Multi-factor authentication · Two-server architecture

1 Introduction

With the rise of mobile application, people are increasingly inseparable from mobile devices. Mobile devices are very useful in our life and become an indispensable part of people. According to the survey, the number of mobile device users in the world reached 6.8 billion in 2019 and is expected to rise to 7.33 billion by 2023 [1], which indicates that everyone in the world is using mobile devices on average. Users can use mobile devices for mobile office, cloud payment, video chat, e-commerce, mobile banking, etc. Mobile communication has penetrated into all aspects of social life.

However, due to the open characteristics of wireless network, although mobile devices bring people convenient and quick experience, there are serious security risks. For example, user privacy information leakage, mobile device loss or theft

Y. Xie—Supported by the National Natural Science Foundation of China under Grant 61862052.

Q. Jiang et al. (Eds.): SPNCE 2022, LNICST 496, pp. 48–61, 2023.
https://doi.org/10.1007/978-3-031-30623-5_4

exposed sensitive information, the hazard of man-in-the-middle attack, etc [2]. Therefore, reliable authentication between the mobile user and the server is essential. Moreover, anonymity in mobile communication and privacy protection of mobile users are indispensable. At present, security authentication protocols with privacy protection function have become the focus of many security researchers.

In [3], an anonymous authentication scheme has been proposed for wireless communication based on a smart card. However, as described in [4], this scheme failed to provide backward security and perfect anonymity. Later, Mun et al. [5] proposed a new global mobile network authentication scheme. Unfortunately, Reddy et al. [6] pointed out that the scheme in [5] was prone to impersonation attack, replay attack and internal privilege attack. To provide a location based service, Memon et al. [7] designed an anonymous communication scheme. But, Reddy et al. [8] found that the protocol still had some security flaws. In distributed systems, single-factor authentication and two-factor authentication are difficult to resist dictionary attack. In [9], a provably secure three-factor scheme has been proposed. Qiu et al. [10] found that the scheme in [9] cannot resist user impersonation attack of key leakage and offline password guessing attack. In addition, the current authentication schemes almost adopt the single-server authentication mode, which have serious security risks and are vulnerable to internal privilege attack and offline dictionary attack, and so on. This is one of the important elements causing the endless security events.

To solve the above problems, we propose an anonymous and practical multi-factor authentication protocol for mobile devices using two-server architecture with honeywords. The protocol uses password, smart card and biometric for identity authentication. Only when the password and biometric factor are correct, the smart card can be activated for identity authentication. It has higher security than single factor and two factors. At the same time, we use a two-server architecture to solve the serious security problems such as internal privilege attacks on a single server and direct leakage of private data after a single server is compromised. Our protocol is demonstrated to be secure under the random oracle model. The performance analysis proves that our proposed scheme is practical and efficient in mobile communication environment.

2 Related Work

Li et al. [11] proposed a user authentication scheme based on biometrics and smart cards. But, Das et al. [12] found that the scheme in [11] has defects in the login and authentication phase, password update phase, and the use of hash function to verify biometric technology. To address these defects, they designed a new scheme, which has good performances such as freely changeable password, low computation cost, and mutual authentication. But then An et al. [13] found that the scheme in [12] cannot resist some security attacks, including impersonation attack, dictionary guessing attack, insider attack, and designed a new scheme. Cao et al. [14] found that the scheme in [13] is prone to replay attack, user masquerading attack and does not guarantee anonymity. They added anonymity to

the improved scheme and claimed that the scheme can resist many attacks. Park et al. [15] found that the scheme in [14] can not resist dictionary guessing attack and server impersonation attack. Then, they designed a new scheme. However, the scheme was later found to have many security problems.

Tan et al. [16] proposed a three-factor authentication scheme for user anonymity in the medical system. Unfortunately, Arshad et al. [17] found that the scheme in [16] could not resist replay attack, they designed a new scheme to solve this defect. Lu et al. [18] found that the scheme in [17] could not resist offline dictionary guessing attack and user impersonation attack. Then, they proposed a new scheme. Amin et al. [19] found that Arshad et al.'s scheme could not resist identity trace attack, impersonation attack and designed a new scheme. Wazid et al. [20] found that the scheme in [19] could not resist internal privilege attack, and proposed an efficient three-factor user authentication and key agreement scheme. However, it can be found that the scheme does not achieve true three-factor security.

3 Background

3.1 Fuzzy Extractor

In order to extract secure cryptographic key from biometric information, the fuzzy extractor is proposed [21]. The fuzzy extractor has generation algorithm and regeneration algorithm, described as follows: (1) $\langle \sigma, \theta \rangle = Gen\,(BIO)$. This is the generation algorithm, the fuzzy extractor inputs BIO based on biometric to generate a uniformly random string σ and an auxiliary string θ. (2) $\sigma = Rep\,(BIO', \theta)$. This is the regeneration algorithm because there are small deviations when sampling the same biometric BIO' (e.g., fingerprints) from the same person at different times. If $dis\,(BIO', BIO) \leq t$ (t is a parameter), that is, BIO' and BIO are fairly close to each other, the fuzzy extractor can recover σ by entering BIO' and the public auxiliary string θ.

4 System Model

Our system model is shown in Fig. 1, there are three types of entities involved in our system, namely the user (U_i), the primary server ($Server_A$) and the secondary server ($Server_B$) in the two-server model. In the registration stage, the user U_i divides his secret value into two parts, and sends his personal data and part of the secret value to the two servers respectively for registration. Each server has only part of the secret value, and $Server_A$ securely sends a smart card with some parameters to U_i. U_i inserts the smart card into the card reader. Then, U_i inputs identity ID_i, password PW_i and biometric BIO_i^* to authenticates with the two servers. Finally, $Server_A$ and U_i generate session key for subsequent secure communication.

User: U_i can send a login request to the system. After the system passes the authentication, the login is successful. Before logging in, U_i should register on the two servers respectively, which enables the server to verify the request sent by U_i.

Server: primary server $Server_A$ and secondary server $Server_B$. $Server_B$ performs the auxiliary calculation, and then $Server_A$ verifies whether the login request of U_i is legitimate. The servers cannot obtain any valid user information except the ciphertext.

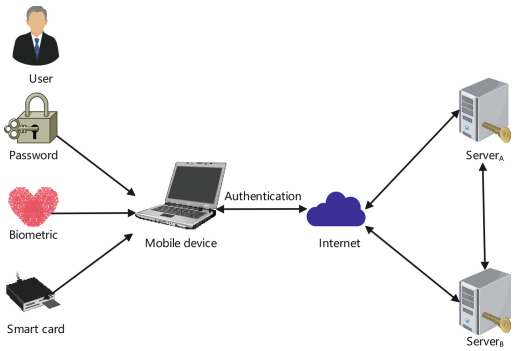

Fig. 1. System model.

5 Our Proposed Protocol

To simplify the following description, we give some notations: $\oplus$ indicates XOR and $\|$ indicates string concatenation.

5.1 System Initialization Phase

G represents a cyclic group whose generator is g, and q is prime order. $h(\cdot)$, $h_0(\cdot)$ are hash functions with ranges $\{0,1\}^l$ (l denotes output length of hash function).

5.2 Registration Phase

The U_i will register separately on two designated servers through a secure channel.

Step 1: U_i selects an identity ID_i and a password PW_i, U_i inputs biometric BIO_i and uses $Gen(\cdot)$ function to generate secret key σ_i and public parameter θ_i, where $\langle \sigma_i, \theta_i \rangle = Gen(BIO_i)$. U_i randomly generates an integer $2^4 \leq n_0 \leq 2^8$ and calculates $\alpha_i = h(h(ID_i \| \sigma_i) \bmod n_0)$. U_i generates random numbers

$x_1 \in Z_q^*$, $\beta_i \in Z_q^*$ and calculates x_2 to satisfy $\alpha_i\beta_i \leftarrow x_1x_2 \,(\mathrm{mod}q)$. Then, U_i sends $\{ID_i, x_1\}$ to the $Server_A$ using a secure communication channel. Similarly, sending $\{ID_i, x_2\}$ to the $Server_B$ using a secure communication channel.

Step 2: After getting $\{ID_i, x_1\}$, $Server_A$ generates the private and public keys (sk_1, PK_1), $PK_1 = g^{sk_1}$, where $sk_1 \in Z_q^*$. $Server_A$ chooses a random integer $e_i \in Z_q^*$ and calculates $A_i = h\,(ID_i \,\|sk_1\,\|e_i\,)$. $Server_A$ stores $\{ID_i, x_1, e_i, Honey\_List = 0\}$ in its database, issues a smart card SC_i to U_i, where $SC_i = \{A_i, h(\cdot)\}$.

Step 3: Upon getting $\{ID_i, x_2\}$ from U_i, $Server_B$ generates the private and public keys (sk_2, PK_2), $PK_2 = g^{sk_2}$, where $sk_2 \in Z_q^*$. $Server_B$ stores $\{ID_i, x_2\}$ in its database.

Step 4: Upon getting $SC_i = \{A_i, h(\cdot)\}$ from $Server_A$, U_i calculates $B_i = h\,((h\,(ID_i) \oplus \alpha_i \oplus h\,(PW_i)) \bmod n_0)$ and $A_i^* = A_i \oplus B_i \oplus \sigma_i$. Finally, U_i replaces A_i with A_i^* in SC_i, and SC_i contains parameters: $SC_i = \{A_i^*, B_i, \theta_i, \beta_i, n_0, h\,(\cdot)\,, h_0\,(\cdot)\,, Gen\,(\cdot)\,, Rep\,(\cdot)\}$.

Remark 1. Generally, the contents of $Honey\_List$ are made up of all the honeywords that the attacker has tried, which look like real passwords but are not [22]. In our paper, all the honeywords A_i' that the attacker has tried are in the $Honey\_List$. These honeywords look like real parameters $A_i = h\,(ID_i \,\|sk_1\,\|e_i\,)$ but are not.

5.3 Login and Authentication Phase

The registered U_i can make a login request, and the two server cooperate to verify whether the login request is valid.

Step 1: U_i inserts SC_i into the card reader. Then, U_i inputs identity ID_i, password PW_i and biometric BIO_i^*. SC_i computes $\sigma_i = Rep\,(BIO_i^*, \theta_i)$, $\alpha_i = h\,(h\,(ID_i \,\|\sigma_i\,) \bmod n_0)$ and verifies $B_i \stackrel{?}{=} h\,((h\,(ID_i) \oplus \alpha_i \oplus h\,(PW_i)) \bmod n_0)$. If verification fails, SC_i refuses the login request. If successful, SC_i calculates $A_i = A_i^* \oplus B_i \oplus \sigma_i$. Then, SC_i generates a random integer $r_i \in Z_q^*$ and current timestamp TS_1. SC_i calculates $R_0 = g^{r_i} \,(\mathrm{mod}q)$, $C_0 = ID_i \oplus h\,(PK_1^{r_i})$, $C_1 = A_i \oplus h\,(ID_i \,\|TS_1\,)$,$C_2 = h(ID_i \,\|A_i\,\|C_1\,\|R_0\,\|TS_1\,)$. Finally, SC_i sends $\{R_0, C_0, C_1, C_2, TS_1\}$ to $Server_A$.

Step 2: After obtaining the message $\{R_0, C_0, C_1, C_2, TS_1\}$, $Server_A$ judges $|TS_1 - TS_1^*| \leq \Delta T$?, where TS_1^* indicates the time when the message was received and ΔT indicates the maximum transmission delay. If the verification is successful, $Server_A$ calculates $ID_i' = C_0 \oplus h\left(R_0^{sk_1}\right)$. Then, $Server_A$ searches $\{ID_i, x_1, e_i, Honey\_List\}$ in its database. If ID_i' (ID_i' is used to distinguish between the ID stored in the database and the ID' calculated by the server) is not found, the authentication process is stopped. Otherwise, $Server_A$ computes $A_i' = C_1 \oplus h\,(ID_i \,\|TS_1\,)$, $A_i = h\,(ID_i \,\|sk_1\,\|e_i\,)$ and judges whether the derived A_i' equals the computed A_i. If not, $Server_A$ learns that the smart card of U_i has been corrupted and the real password was not obtained by the adversary. Therefore, $Server_A$ adds A_i' to $Honey\_List$ and ignores the login

request. In addition, if $|Honey\_List| \geq S_0$, where S_0 represents a threshold, $Server_A$ pauses using the smart card. If they are equal, $Server_A$ checks whether $C_2 \stackrel{?}{=} h(ID_i \| A_i \| C_1 \| R_0 \| TS_1)$. If it does not match, the authentication process is stopped. Otherwise, $Server_A$ generates current timestamp TS_2 and computes $R_1 = R_0^{x_1} (\mathrm{mod} q)$, $C_3 = R_1 \oplus h_0\left(PK_2^{sk_1}\right)$, $C_4 = ID_i \oplus h\left(PK_2^{sk_1} \| TS_2\right)$, $C_5 = h(ID_i \| C_4 \| R_1 \| TS_2)$. Finally, $Server_A$ sends $\{C_3, C_4, C_5, TS_2\}$ to $Server_B$.

Step 3: After receiving $\{C_3, C_4, C_5, TS_2\}$, $Server_B$ checks the condition $|TS_2 - TS_2^*| \leq \Delta T$?. If the verification is successful, $Server_B$ computes $ID_i' = C_4 \oplus h\left(PK_1^{sk_2} \| TS_2\right)$. Then, $Server_B$ searches $\{ID_i, x_2\}$ in its database. If ID_i' (ID_i' is used to distinguish between the ID stored in the database and the ID' calculated by the server) is not found, the authentication process is terminated. Otherwise, $Server_B$ computes $R_1 = C_3 \oplus h_0\left(PK_1^{sk_2}\right)$ and checks whether $C_5 \stackrel{?}{=} h(ID_i \| C_4 \| R_1 \| TS_2)$. If it does not match, the session is terminated. Otherwise, $Server_B$ generates current timestamp TS_3 and computes $R_2 = R_1^{x_2} (\mathrm{mod} q)$, $C_6 = R_2 \oplus h_0\left(PK_1^{sk_2}\right)$, $C_7 = h(ID_i \| R_2 \| TS_3)$. Finally, $Server_B$ sends $\{C_6, C_7, TS_3\}$ to $Server_A$.

Step 4: After receiving the message $\{C_6, C_7, TS_3\}$, $Server_A$ checks the condition $|TS_3 - TS_3^*| \leq \Delta T$?. If the verification holds, $Server_A$ computes $R_2 = C_6 \oplus h_0\left(PK_2^{sk_1}\right)$ and judges whether $C_7 \stackrel{?}{=} h(ID_i \| R_2 \| TS_3)$. If the match fails, the session is terminated. Otherwise, $Server_A$ generates current timestamp TS_4 and calculates $SK_s = h(ID_i \| A_i \| R_0^{sk_1} \| R_2)$, $C_8 = h(ID_i \| A_i \| R_2 \| SK_s \| TS_4)$. Finally, $Server_A$ sends $\{C_8, TS_4\}$ to U_i.

Step 5: Upon getting the message $\{C_8, TS_4\}$, U_i judges $|TS_4 - TS_4^*| \leq \Delta T$?. If the verification is successful, SC_i continues to compute $R_2 = R_0^{\alpha_i \beta_i} (\mathrm{mod} q)$, $SK_u = h(ID_i \| A_i \| PK_1^{r_i} \| R_2)$ and checks $C_8 \stackrel{?}{=} h(ID_i \| A_i \| R_2 \| SK_u \| TS_4)$. If it does not match, the session is terminated. Otherwise, both SC_i and $Server_A$ generate the common session key $SK = SK_u = SK_s$.

5.4 Password and Biometric Update Phase

This phase provides user password and biometric update functionality. This process is performed locally and does not require communication with the servers.

Step 1: U_i inserts SC_i into the card reader. Then, U_i provides ID_i, old PW_i and current biometric BIO_i^*. SC_i calculates $\sigma_i = Rep(BIO_i^*, \theta_i)$, $\alpha_i = h(h(ID_i \| \sigma_i) \bmod n_0)$ and judges $B_i \stackrel{?}{=} h((h(ID_i) \oplus \alpha_i \oplus h(PW_i)) \bmod n_0)$. SC_i refuses this update request if the match fails. Otherwise, SC_i requires U_i to supply new password and input new biometric.

Step 2: U_i inputs a new PW_i^{new} and a new BIO_i^{new}. Then, SC_i generates a random integer $2^4 \leq n_0' \leq 2^8$ and calculates $\langle \sigma_i^{new}, \theta_i^{new} \rangle = Gen(BIO_i^{new})$, $\alpha_i^{new} = h(h(ID_i \| \sigma_i^{new}) \bmod n_0')$. SC_i computes $\beta_i^{new} \in Z_q^*$ to satisfy $\alpha_i^{new} \beta_i^{new} = \alpha_i \beta_i$, $B_i^{new} = h((h(ID_i) \oplus \alpha_i^{new} \oplus h(PW_i^{new})) \bmod n_0')$, $A_i^{*new} = A_i^* \oplus B_i \oplus \sigma_i \oplus B_i^{new} \oplus \sigma_i^{new}$.

Step 3: Finally, SC_i replaces $\{A_i^*, B_i, \theta_i, \beta_i, n_0\}$ with $\{A_i^{*new}, B_i^{new}, \theta_i^{new}, \beta_i^{new}, n_0'\}$.

5.5 Smart Card Revocation Phase

If a legitimate U_i lost SC_i, U_i can apply for a new smart card SC_i^{new}. The specific process is as follows:

Step 1: U_i uses the original ID_i, and selects a new PW_i'. U_i inputs biometric BIO_i' and computes $\langle \sigma_i', \theta_i' \rangle = Gen\left(BIO_i'\right)$. U_i generates a random number $2^4 \leq n_0' \leq 2^8$ and computes $\alpha_i' = h\left(h\left(ID_i \| \sigma_i'\right) \bmod n_0'\right)$. U_i generates random numbers $x_1' \in Z_q^*$, $\beta_i' \in Z_q^*$ and computes x_2' to satisfy $\alpha_i'\beta_i' \leftarrow x_1'x_2' (\mathrm{mod} q)$. Then, U_i sends $\{ID_i, x_1'\}$ to the $Server_A$ using a secure channel. Similarly, sending $\{ID_i, x_2'\}$ to the $Server_B$ using a secure channel.

Step 2: Upon getting $\{ID_i, x_1'\}$ from U_i, $Server_A$ generates the private and public keys (sk_1, PK_1), $PK_1 = g^{sk_1}$, where $sk_1 \in Z_q^*$. $Server_A$ chooses a random value $e_i' \in Z_q^*$ and calculates $A_i' = h\left(ID_i \| sk_1 \| e_i'\right)$. $Server_A$ stores $\{ID_i, x_1', e_i', Honey\_List = 0\}$, issues a new $SC_i^{new} = \{A_i', h(\cdot)\}$ to U_i.

Step 3: Upon getting $\{ID_i, x_2'\}$ from U_i, $Server_B$ generates the private and public keys (sk_2, PK_2), $PK_2 = g^{sk_2}$, where $sk_2 \in Z_q^*$. $Server_B$ stores $\{ID_i, x_2'\}$ in its database.

Step 4: After receiving SC_i^{new} from $Server_A$, the user U_i computes $B_i' = h\left(\left(h\left(ID_i\right) \oplus \alpha_i' \oplus h\left(PW_i{}'\right)\right) \bmod n_0'\right)$ and $A_i^{*\prime} = A_i{}' \oplus B_i' \oplus \sigma_i{}'$. Then, U_i replaces $A_i{}'$ with $A_i^{*\prime}$ in SC_i^{new}. Finally, $SC_i{}^{new} = \{A_i^{*\prime}, B_i{}', \theta_i{}', \beta_i{}', n_0', h\left(\cdot\right), h_0\left(\cdot\right), Gen\left(\cdot\right), Rep\left(\cdot\right)\}$.

6 Security Analysis

6.1 Security Model

We propose a formal security model under the assumption of CDHP, which is mainly inspired by the model for AKE [23,24].

Participants and Initialization. Our scheme involves three participants: U_i, $Server_A$, $Server_B$. We define the i-th instance of U_i as U^i, the j-th instance of $Server_A$ as S_A^j, and the k-th instance of $Server_B$ as S_B^k. We define P^i as the i-th instance of a participant. Each user U_i selects a password PW_{U_i} from the dictionary $\mathcal{D}$.

Queries. Through oracle queries, the adversary $\mathcal{A}$ interacts with the simulator $\mathcal{C}$. Oracle queries simulate $\mathcal{A}$'s ability in true attacks. $\mathcal{A}$ can use the following query types:

- $Execute\left(U^i, S_A^j, S_B^k\right)$: We use the query to simulate $\mathcal{A}$'s passive eavesdropping attack. $\mathcal{A}$ can run this query to obtain the messages interchange among the three participants U^i, S_A^j and S_B^k.
- $Send\left(P^i, M\right)$: The active attack of $\mathcal{A}$ will be simulated by using the query. $\mathcal{A}$ simulates a participant sending message M to P^i and gets the response from P^i.

- *Reveal* $\left(P^i\right)$: We use this query to simulate a known session key attack. If the instance P^i accepts the session, the session key SK held by P^i is returned. Otherwise, the response is NULL.
- *Corrupt* $\left(U^i, a\right)$: We use this query to simulate the corruption ability of $\mathcal{A}$. If $a = 1$, then PW of U_i is displayed; if $a = 2$, then it reveals all parameters stored in the smart card.
- *Corrupt* $\left(S_A^j / S_B^k, a\right)$: The private key of server is displayed if $a = 1$. The account $\{ID_i, x_2\}$ is printed if $a = 2$.
- $DDHP$: The decision DHP oracle is to check $\left(g^a, g^b\right)? = g^{ab}$, only $\mathcal{C}$ can query the oracle once in one session.
- *Test* $\left(P^i\right)$: This query will not simulate any actual functionality of $\mathcal{A}$. It can define semantic security. If P^i accepts, flip a coin b. If $b = 1$, the real SK is sent to $\mathcal{A}$. Otherwise, $\mathcal{A}$ will be given a random value (the length is the same as SK). Whether the key is real or random is something that $\mathcal{A}$ must determine.

Partner. Two instances U^i and S_A^j are partners of each other if they meet the following conditions: (1) U^i and S_A^j are accepted (this indicates that SK has been negotiated). (2) The same session identification *sid* is shared between U^i and S_A^j. (3) U^i's partner identifier *pid* is S_A^j and vice versa.

Freshness. If the instance is fresh, the following three conditions need to be met: (1) P has calculated an accepted SK. (2) The Reveal queries are not made by P and its partner. (3) P is asked the Corrupt queries at most only once.

Semantic security of session key. $\mathcal{A}$ needs to distinguish between a real SK and a random key. $\mathcal{A}$ can make several Test queries for some fresh instances. b' is the guessed bit for bit b selected for the Test query. If the condition $b' = b$ is met, $\mathcal{A}$ wins the game, which is expressed as $Succ$. For protocol P, the advantage of $\mathcal{A}$ for destroying its semantic security is:

$$Adv_P^{AKA}\left(\mathcal{A}\right) = \left|2Pr\left[Succ\left(\mathcal{A}\right)\right] - 1\right| = \left|2Pr\left[b' = b\right] - 1\right|. \tag{1}$$

6.2 Security Proof

Theorem 1. *For protocol P, it is assumed that $\mathcal{A}$ breaches the semantic security within a bounded time t in the random oracle, $\mathcal{D}$ is a uniformly distributed dictionary, and the size of $\mathcal{D}$ is $|\mathcal{D}|$. Suppose that $\mathcal{A}$ asks less than q_s times sessions, q_d times Send queries, q_e times Execute queries and q_h times Hash oracle queries. Then,*

$$\begin{aligned} Adv_P^{AKA}\left(\mathcal{A}\right) \leq \frac{q_h^2}{2^{l+2}} + \frac{\left(q_d + q_e\right)^2}{2q} + 4\frac{q_d}{2^l} + \frac{q_d}{|\mathcal{D}|} + \\ q_s^2 q_h Adv_G^{DLP}\left(t\right) + q_s q_h Adv_G^{CDH}\left(t + \left(q_d + q_e\right) t_e\right). \end{aligned} \tag{2}$$

where t_e represents the computation time of exponentiation in G.

Proof. Suppose that $\mathcal{A}$ breaches the semantic security of P, $\mathcal{C}$ can solve $CDHP$ problem. The proof consists of sequence of games. For each $Game_i$, we assume that $\mathcal{A}$ correctly guesses b is an event $Succ_i$. $AskPara_n$ represents $\mathcal{A}$ calculating A_i by Hash queries with $(ID_i \| sk_1 \| e_i)$. $AskH_n$ represents $\mathcal{A}$ calculating A_i by Hash queries with $(D_i \| A_i \| R_2 \| SK_s \| TS_4)$.

$Game_0$: This game corresponds to a true attack without any restrictions, where $Succ_0 = \{b' = b\}$. Thus,

$$Adv_P^{AKA}(\mathcal{A}) \leq |2Pr[Succ_0] - 1|. \tag{3}$$

$Game_1$: $\mathcal{C}$ simulates a random oracle h in this game, a hash list $List_h = (i, j, k)$ is generated (it is empty at the beginning.). When $\mathcal{A}$ runs a query j, if the request has been asked before, it will give the answer k from the list $List_h$. Otherwise, $\mathcal{C}$ chooses $j \in_n (0,1)^l$ and returns the answer j, while adding the new record (i, j, k) to $List_h$, where i stands for the query time, j stands for the content set, and k stands for the corresponding answer set.

Compared to $Game_0$, $\mathcal{C}$ sets a relevant record in $Game_1$, so this game is completely indistinguishable from the true game. Thus,

$$|Pr[Succ_1] - Pr[Succ_0]| = 0. \tag{4}$$

$Game_2$: This game simulates all queries, just like in the $Game_1$. If a collision occurs, all simulations will be aborted. The probability of collision on the output of hash queries is at most $\frac{q_h^2}{2^{l+1}}$. The probability of collision on the exchanged messages $\{R_0, C_0, C_1, C_2, TS_1\}$, $\{C_3, C_4, C_5, TS_2\}$, $\{C_6, C_7, TS_3\}$ and $\{C_8, TS_4\}$ is at most $\frac{(q_d+q_e)}{2q}^2$. On the basis of the birthday paradox, we can get,

$$|Pr[Succ_2] - Pr[Succ_1]| \leq \frac{q_h^2}{2^{l+1}} + \frac{(q_d + q_e)^2}{2q}. \tag{5}$$

$Game_3$: Without random oracle query, the execution are halted if $\mathcal{A}$ correctly guesses the authentication elements C_2, C_5, C_7, C_8. The game is indistinguishable from $Game_2$, except that the instance refuses to use the correct authentication elements. Hence,

$$|Pr[Succ_3] - Pr[Succ_2]| \leq \frac{q_d}{2^l}. \tag{6}$$

$Game_4$: If $\mathcal{A}$ guessed A_i without the Hash query and successfully spoofed U_i or servers, then execution will pause. Therefore, we can get,

$$|Pr[Succ_4] - Pr[Succ_3]| \leq \frac{q_d}{2^l}. \tag{7}$$

$Game_5$: The executions are finished if $\mathcal{A}$ calculates the secret value A_i by Hash queries with $(ID_i \| sk_1 \| e_i)$. $Game_5$ and $Game_4$ are indistinguishable if $AskPara_5$ does not occur. Therefore,

$$|Pr[Succ_5] - Pr[Succ_4]| \leq Pr[AskPara_5] \tag{8}$$

From security model, $\mathcal{A}$ can query both $Corrupt\left(U^i, a\right)$ and $Corrupt\left(S_A^j/S_B^k, a\right)$. Therefore, we obtain the following conclusions.

$$\begin{aligned}
&Pr\left[AskPara_5\right] = \\
&Pr\left[AskPara_5 WithCorrupt\left(U_i, 1\right)\right] \\
&+Pr\left[AskPara_5 WithCorrupt\left(U_i, 2\right)\right] \\
&+Pr\left[AskPara_5 WithCorrupt\left(S_A^j/S_B^k, 1\right)\right] \\
&+Pr\left[AskPara_5 WithCorrupt\left(S_A^j/S_B^k, 2\right)\right] \\
&= Pr\left[* \| PW_i\right] + Pr\left[\sigma_i \| *_i\right] + Pr\left[ID_i \| * \| PW_i\right] + Pr\left[ID_i \| * \| sk_1\right] \\
&\leq \tfrac{q_d}{2^l} + \tfrac{q_d}{|\mathcal{D}|} + \tfrac{q_d}{2^l} + q_s^2 q_h Adv_G^{DLP}\left(t\right) = 2\tfrac{q_d}{2^l} + \tfrac{q_d}{|\mathcal{D}|} + q_s^2 q_h Adv_G^{DLP}\left(t\right)
\end{aligned} \tag{9}$$

$Game_6$: The execution are halted if $\mathcal{A}$ calculates the values (C_2, C_5, C_7, C_8, SK). If $AskH_6$ occurs, $\mathcal{A}$ queries the hash function with $(ID_i \| A_i \| R_0^{sk_1} \| R_2)$, where $CDHP\ (R_0, PK_1) = R_0^{sk_1}$. $\mathcal{C}$ selects a session as the test session and adds the $CDHP$ parameter, $\mathcal{C}$ can solve the CDHP.

In a non-test session, $\mathcal{A}$ makes a Hash query with (C_2, C_5, C_7, C_8). $\mathcal{C}$ receives such a query will check the list $List_h$. If the request was asked before, it will give the same answer. DDHP oracle to check $(R_0, PK_1)? = R_0^{sk_1}$. If it is not equal, NULL is returned, otherwise a random value to $\mathcal{A}$ is returned, due to $\mathcal{C}$ does not know r_i and sk_1 in (C_2, C_5, C_7, C_8).

In a test session, $\mathcal{C}$ randomly selects g^x and g^y, lets $R_0 = g^x$, $PK_1 = g^y$. $\mathcal{A}$ makes a Hash query with $(ID_i \| A_i \| g^{xy} \| R_2)$, where $g^{xy} = CDHP(R_0, PK_1)$. This game is no different from the previous one if $AskH_6$ does not occur. Therefore,

$$\left|Pr\left[Succ_6\right] - Pr\left[Succ_5\right]\right| \leq Pr\left[AskH_6\right]. \tag{10}$$

where, $Pr\left[AskH_6\right] \leq q_s q_h Adv_G^{CDH}\left(t + (q_d + q_e)t_e\right)$, $Pr\left[Succ_6\right] = \frac{1}{2}$.

Finally, We can conclude that,

$$\begin{aligned}
&Adv_P^{AKA}\left(\mathcal{A}\right) = 2Pr\left[Succ_0\right] - 1 \\
&= 2Pr\left[Succ_6\right] - 1 + 2\left(Pr\left[Succ_0\right] - Pr\left[Succ_6\right]\right) \\
&\leq 2\{\left|Pr\left[Succ_1\right] - Pr\left[Succ_0\right]\right| + \left|Pr\left[Succ_2\right] - Pr\left[Succ_1\right]\right| \\
&+ \left|Pr\left[Succ_3\right] - Pr\left[Succ_2\right]\right| + \left|Pr\left[Succ_4\right] - Pr\left[Succ_3\right]\right| \\
&+ \left|Pr\left[Succ_5\right] - Pr\left[Succ_4\right]\right| + \left|Pr\left[Succ_6\right] - Pr\left[Succ_5\right]\right|\} \\
&\leq \tfrac{q_h^2}{2^{l+1}} + \tfrac{(q_d+q_e)^2}{2q} + 4\tfrac{q_d}{2^l} + \tfrac{q_d}{|\mathcal{D}|} + q_s^2 q_h Adv_G^{DLP}\left(t\right) \\
&+ q_s q_h Adv_G^{CDH}\left(t + (q_d + q_e)\, t_e\right).
\end{aligned} \tag{11}$$

□

7 Performance Analysis

We analyze the security features, computation costs and the communication costs of our scheme and several related schemes. For a fair comparison, we choose several authentication protocols with two-server architecture.

7.1 Security Feature

Table 1 shows the comparison between the security features of our scheme and the four related schemes [25–28]. We define $SF1$, $SF2$, $SF3$, $SF4$, $SF5$, $SF6$, $SF7$ and $SF8$ are user anonymity, forward secrecy, offline password guessing attack, freely password update, insider attack, key agreement, online password guessing attack and formal security proof, respectively. Where, ✓indicates that the scheme meets relevant security requirements. × indicates that the scheme does not meet the security requirements. From Table 1, it can be see that our scheme meets all the security requirements.

Table 1. Security features comparison.

Schemes	$SF1$	$SF2$	$SF3$	$SF4$	$SF5$	$SF6$	$SF7$	$SF8$
Kumari et al. [25]	×	✓	✓	×	✓	✓	×	✓
Yi et al. [26]	×	×	✓	×	×	✓	✓	✓
Jin et al. [27]	×	×	✓	×	×	✓	×	×
Zhang et al. [28]	×	✓	×	×	✓	✓	×	✓
Our	✓	✓	✓	✓	✓	✓	✓	✓

7.2 Computation Cost

The various notations and the running times of the existing experiments [29, 30] are shown in Table 2. The experimental equipment is a four-core 3.2 GHz machine with 8 GB memory. We ignore lightweight operations. Table 3 shows the comparison of client and server computation costs between our scheme and other schemes in the login and authentication phase.

In our scheme, the computation cost is 295.1 millliseconds. In [25–28], the computation costs of their schemes are 390.88, 768.16, 473.14, 472.8 milllisecconds, respectively. Through the above results, our computation cost is much lower compared to similar schemes, which proves the absolute computational advantage of our two-server scheme.

Table 2. The execution time of related operations..

Notation	Operations	Time(in seconds)
T_e	Exponentiation operation	0.0192
T_m	Modular multiplication operation	0.00088
T_h	Regular hash operation	0.00032
T_f	Biometric-fuzzy extractor operation	0.0171

Table 3. Computation costs comparison.

Protocols	Client	Server	Total cost
Kumari et al. [25]	$3T_e + 3T_h$	$17T_e + 2T_h + 6T_m$	$20T_e + 5T_h + 6T_m \approx 390.88ms$
Yi et al. [26]	$19T_e + 5T_h + 10T_m$	$20T_e + 6T_h + 8T_m$	$39T_e + 11T_h + 18T_m \approx 768.16ms$
Jin et al. [27]	$8T_e + 3T_h + 2T_m$	$16T_e + 5T_h + 9T_m$	$24T_e + 8T_h + 11T_m \approx 473.04ms$
Zhang et al. [28]	$8T_e + 5T_h + 2T_m$	$16T_e + 5T_h + 8T_m$	$24T_e + 10T_h + 10T_m \approx 472.8ms$
Our	$4T_e + 10T_h + T_f + T_m$	$10T_e + 16T_h$	$14T_e + 26T_h + T_f + T_m \approx 295.1ms$

7.3 Communication Cost

Table 4 compares the communication costs of different existing schemes and our scheme in the login and authentication phase. We make reasonable assumptions about the length of the security parameter: $l_u = 64$ bits (user identity), $l_r = 128$ bits (random number), $l_h = 256$ bits (SHA-256), $l_t = 32$ bits (timestamp) and $l_g = 1024$ bits (a group element in G).

In our protocol, the communication costs of messages $\{R_0, C_0, C_1, C_2, TS_1\}$, $\{C_3, C_4, C_5, TS_2\}$, $\{C_6, C_7, TS_3\}$ and $\{C_8, TS_4\}$ are 1824 bits, 1568 bits, 1312 bits and 288 bits. Thus, our communication cost is $(1824 + 1568 + 1312 + 288) = 4992$ bits. From Table 4, it can be seen that our scheme is much lower communication cost compared to similar schemes. On the premise of ensuring security, our protocol is more efficient, which is very suitable for mobile lightweight devices with limited resources.

Table 4. Communication costs comparison.

Protocols	Total messages	Total cost (bits)
Kumari et al. [25]	6	12928
Yi et al. [26]	6	22784
Jin et al. [27]	6	12224
Zhang et al. [28]	9	21184
Our	4	4992

8 Conclusion

In this paper, we propose an anonymous and practical multi-factor authentication protocol for mobile devices using two-server architecture with honeywords, which solves the security and privacy preserving problems of user data currently faced by mobile devices. Only when the password and biometric factors are correct, the protocol can activate the smart card for authentication, which has higher security than single factor and two factors. At the same time, we

use a two-server architecture to avoid serious security problems such as internal privilege attack and direct leakage of private data after a single server is compromised. Through a strict security analysis, we demonstrate the security of the protocol. Our performance evaluation proves that our proposed protocol has obvious efficiency advantages compared with similar protocols. Although our protocol meets many security requirements, more functional features are necessary for authentication protocols. Our future work is to increase the functionality of the protocol while keeping it secure and efficient.

References

1. Statista: Forecast number of mobile users worldwide from 2019 to 2023 (2020)
2. Koved, L., Trewin, S., Swart, C., Singh, K., Cheng, P.C., Chari, S.: Perceived security risks in mobile interaction. In: Symposium on usable privacy and security (SOUPS), pp. 24–26 (2013)
3. Zhu, J., Ma, J.: A new authentication scheme with anonymity for wireless environments. IEEE Trans. Consum. Electron. **50**(1), 231–235 (2004)
4. Lee, C.C., Hwang, M.S., Liao, I.E.: Security enhancement on a new authentication scheme with anonymity for wireless environments. IEEE Trans. Industr. Electron. **53**(5), 1683–1687 (2006)
5. Mun, H., Han, K., Lee, Y.S., Yeun, C.Y., Choi, H.H.: Enhanced secure anonymous authentication scheme for roaming service in global mobility networks. Math. Comput. Model. **55**(1–2), 214–222 (2012)
6. Goutham Reddy, A., Yoon, E.J., Das, A.K., Yoo, K.Y.: Lightweight authentication with key-agreement protocol for mobile network environment using smart cards. IET Inf. Secur. **10**(5), 272–282 (2016)
7. Memon, I., Hussain, I., Akhtar, R., Chen, G.: Enhanced privacy and authentication: an efficient and secure anonymous communication for location based service using asymmetric cryptography scheme. Wireless Pers. Commun. **84**(2), 1487–1508 (2015)
8. Reddy, A.G., Das, A.K., Yoon, E.J., Yoo, K.Y.: A secure anonymous authentication protocol for mobile services on elliptic curve cryptography. IEEE Access **4**, 4394–4407 (2016)
9. Islam, S.H., Vijayakumar, P., Bhuiyan, M.Z.A., Amin, R., Balusamy, B., et al.: A provably secure three-factor session initiation protocol for multimedia big data communications. IEEE Internet Things J. **5**(5), 3408–3418 (2017)
10. 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 Secure Comput. **19**(2), 1338–1351 (2020)
11. Li, C.T., Hwang, M.S.: An efficient biometrics-based remote user authentication scheme using smart cards. J. Netw. Comput. Appl. **33**(1), 1–5 (2010)
12. Das, A.K.: Analysis and improvement on an efficient biometric-based remote user authentication scheme using smart cards. IET Inf. Secur. **5**(3), 145–151 (2011)
13. An, Y.: Security analysis and enhancements of an effective biometric-based remote user authentication scheme using smart cards. J. Biomed. Biotechnol. **2012** (2012)
14. Cao, L., Ge, W.: Analysis and improvement of a multi-factor biometric authentication scheme. Secur. Commun. Netw. **8**(4), 617–625 (2015)

15. Park, Y., Park, K., Lee, K., Song, H., Park, Y.: Security analysis and enhancements of an improved multi-factor biometric authentication scheme. Int. J. Distrib. Sens. Netw. **13**(8), 1550147717724308 (2017)
16. Tan, Z.: A user anonymity preserving three-factor authentication scheme for telecare medicine information systems. J. Med. Syst. **38**(3), 1–9 (2014)
17. Arshad, H., Nikooghadam, M.: Three-factor anonymous authentication and key agreement scheme for telecare medicine information systems. J. Med. Syst. **38**(12), 1–12 (2014)
18. Lu, Y., Li, L., Peng, H., Yang, Y.: An enhanced biometric-based authentication scheme for telecare medicine information systems using elliptic curve cryptosystem. J. Med. Syst. **39**(3), 1–8 (2015)
19. Amin, R., Islam, S., Biswas, G., Khan, M.K., Obaidat, M.S.: Design and analysis of an enhanced patient-server mutual authentication protocol for telecare medical information system. J. Med. Syst. **39**(11), 1–20 (2015)
20. Wazid, M., Das, A.K., Kumari, S., Li, X., Wu, F.: Design of an efficient and provably secure anonymity preserving three-factor user authentication and key agreement scheme for tmis. Secur. Commun. Netw. **9**(13), 1983–2001 (2016)
21. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_31
22. Juels, A., Rivest, R.L.: Honeywords: Making password-cracking detectable. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, pp. 145–160 (2013)
23. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_11
24. Liu, X., Li, Y., Qu, J., Jiang, Q.: Maka: provably secure multi-factor authenticated key agreement protocol. J. Internet Technol. **19**(3), 669–677 (2018)
25. Anitha Kumari, K., Sudha Sadasivam, G.: Two-server 3d elgamal diffie-hellman password authenticated and key exchange protocol using geometrical properties. Mobile Netw. Appl. **24**(3), 1104–1119 (2019)
26. Yi, X., Hao, F., Bertino, E.: ID-based two-server password-authenticated key exchange. In: Kutyłowski, M., Vaidya, J. (eds.) ESORICS 2014. LNCS, vol. 8713, pp. 257–276. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11212-1_15
27. Jin, H., Wong, D.S., Xu, Y.: An efficient password-only two-server authenticated key exchange system. In: Qing, S., Imai, H., Wang, G. (eds.) ICICS 2007. LNCS, vol. 4861, pp. 44–56. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77048-0_4
28. Zhang, H., Kumari, S., Obaidat, M.S., Wei, F.S.: Gateway-oriented two-server password authenticated key exchange protocol for unmanned aerial vehicles in mobile edge computing. IET Commun. **14**(15), 2427–2433 (2020)
29. Srinivas, J., Das, A.K., Kumar, N., Rodrigues, J.J.: Cloud centric authentication for wearable healthcare monitoring system. IEEE Trans. Dependable Secure Comput. **17**(5), 942–956 (2018)
30. 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 Secure Comput. **17**(6), 1133–1146 (2018)