Hindawi
Security and Communication Networks
Volume 2019, Article ID 1694702, 12 pages
https://doi.org/10.1155/2019/1694702

Research Article

Fingerprint Protected Password Authentication Protocol

Chao Yang[1,2] **Junwei Zhang**[1] **Jingjing Guo**[1] **Yu Zheng**[1] **Li Yang**[3] **and Jianfeng Ma**[1]

[1]*School of Cyber Engineering, Xidian University, Xi'an 710071, China*
[2]*Science and Technology on Communication Networks Laboratory, Shijiazhuang 050081, China*
[3]*School of Computer, Xidian University, Xi'an 710071, China*

Correspondence should be addressed to Chao Yang; chaoyang@xidian.edu.cn

Received 12 March 2019; Accepted 22 May 2019; Published 26 June 2019

Guest Editor: Fagen Li

With the rapid development of industrial Internet of things (IIOT), a variety of cloud services have been deployed to store and process the big data of IIOT. The traditional password only authentication is unable to meet the needs of security situation in IIOT. Therefore, a lot of mobile phone assisted password authentication schemes have been proposed. However, in existing schemes, the secret information is required to be stored in the user's mobile phone. Once the phone is lost, the secret information may be obtained by the opponent, which will bring irreparable loss to the user. To address the above problems, we propose a fingerprint protected password authentication scheme which has no need to store the secret parameter in the mobile phone. When a user logs in, he uses his mobile phone to generate the private key which is used to decrypt the encrypted text generated during the registration phase. The process of generating the private key needs to enter the password and the fingerprint. When the computer interacts with the mobile phone, the user's password will be blinded so that it can protect the user's password from adversary's attacks. Theoretical analysis and experimental results show that our scheme improves the security of the user's secret. Meanwhile, our scheme can resist the opponent's dictionary attacks, replay attacks, and phishing attack. Our scheme can reduce the storage pressure of the mobile phone and is easy to deploy.

1. Introduction

The rapid development of industrial Internet of things (IIOT) has reformed many aspects of user's daily life. A variety of cloud services have been deployed to store and process the big data of IIOT [1–4]. Due to the openness of public network, these cloud services suffer from a wide range of attacks [5–8]. Text-based passwords remain the dominant authentication technique for various online services and systems, because it is accessible and convenient. Most of them require their own password and the user has to manage multiple passwords to their accounts.

However, password authentication schemes have inherent limitations. Firstly, the user is prone to use a simple password for convenience. If the length of the password is short, its information entropy will be very low. Therefore, the user's password will be easily compromised by attackers. After obtaining the user's password, the opponent can use the password to access the user's account and steal user's private information. In addition, the user may use a single password to authenticate to multiple services. If one of the user's accounts information leaks, it would pose a threat to the security of other accounts. What is worse, users often forget their passwords and try to login via trial-and-error, which means that a malicious online service would learn not only a user's password to that service, but to many other services, possibly also through a cross-site impersonation attack. The low information entropy and the repeated setting of the password are common problems in the process of password setting and password usage. These above problems are also easily exploited by attackers. Moreover, malicious service providers may also launch attacks against the user. For example, the CEO of Facebook allegedly used Facebook login data to access the private mails of some business rivals and journalists in 2004[9]. Hence, protecting users' passwords has become extremely important for individuals.

To avoid the limitation of password authentication mechanism, researchers have proposed many schemes [10–12]. Adding an additional factor is one of the common ways to protect user's password. With the extensive use of mobile phones in recent years, users often use mobile phone to manage personal information and store many credentials or secrets on the mobile phone. Consequently, the research on mobile phone assisted password authentication is one of the important trends.

In 2004, Wu et al. [13] proposed a scheme using the mobile phone as an authentication token with the help of a trusted agent P. When a user wants to authenticate to a remote server, he/she is required to communicate with the agent P first. The agent P sends a session name to the mobile phone and the PC client. The user compares the session name showing on the mobile phone and the PC client. Then the user sends the result to the agent P. However, this scheme requires the agent to be secure and trustworthy, which is a strong assumption in practice. Moreover, the mobile phone is required to interact with the agent several times and is prone to man-in-the-middle attack. In addition, the involvement of a trust agent makes the scheme hard to be deployed.

The scheme [14] proposed by Thanh et al. uses a mobile phone and an authentication server to assist the user during login authentication. In this scheme, with the help of the mobile phone and the authentication server, the client and the server is not required to communicate directly through the public network while sending or receiving credentials. However, the authentication server is utilized to receive the data transmitted by the mobile terminal through the GSM network, which is considered to be an insecure transmission channel. Therefore, the response phase is prone to be attacked by the opponents. When the user uses a mobile phone to send a message to the authentication server through the GSM network, it is not hard for the opponent to hijack the user's message. The adversary can use the hijacked information to authenticate with the server.

Considering the security of information transmission in the authentication phase, the scheme in [15] uses the mobile phone as the intermediary to transmit the user's secrets securely. It uses the QR code and the camera to encrypt the key during the authentication phase. But this scheme requires adding a bar code on the device in advance. If there are plenty of authentication devices or services, many QR codes are required to be predeployed, which is not practical for deployment.

The scheme in [16] is put forward to resist session hijacking and phishing attacks. The client PC is independent to the mobile device and just performs calculations in the scheme, while the mobile phone records the user's password. They share a session key to against session hijacking. However, the mechanism requires the terminal to calculate and establish a pair of public keys with each server. If there are a number of services, the setting will be cumbersome, which will bring much overhead to the mobile phone. What is more, this scheme needs to establish a secure channel between the client PC and server, which greatly reduces the availability of this scheme.

In [17], a method for users to authenticate with the server on an unreliable computer is provided. The main idea is that the mobile phone scans the QR code on the computer, encrypts the QR code, and then uses it to authenticate with the server. However, this scheme requires the trusted mobile phone and the server to set up a shared key in advance, which is difficult to realize in the practice. Moreover, mobile phone and computer need to communicate with the server several times during the authentication phase, which makes this scheme time consuming.

In the SPA (single password authentication) scheme proposed by Acar et al. [18], it requires additional cloud storage to store the user's data and assist the user to register and log in. However, the scheme needs to set a secure channel between the client and the storage when the client sends the blind signature private key to the storage, which is a strong assumption. Furthermore, this scheme also requires a trusted cloud storage to store the ciphertext, which is not readily accessible in practice. Moreover, the user and the cloud storage require a lot of data transmission and interaction during the authentication phase, making time consuming. Acar et al. also proposed a different SPA scheme based on a mobile device. In this scheme, a trusted mobile device is used to assist users to register and log in. The user uses the password to encrypt the authentication key and then store the ciphertext on the trusted mobile phone device. When the user wants to access his account, he inputs his password to decrypt the ciphertext to get his authentication key and then authenticate with the server. But this scheme also requires the mobile device and server to be trusted, and no collusion between the server and mobile phone, which is a very harsh security requirement for mobile devices. In addition, if the mobile device is lost, which is a very common situation, user's ciphertext stored on the mobile device can be cracked offline by adversaries, which will lead to serious consequences.

In the password authentication method based on mobile phone assistance, there is a method based on fingerprint, which attracts much attention. The scheme [19] proposed using a fingerprint to replace the PIN code to authenticate the user's identity. However, fingerprint information is still stored in the mobile phone in this scheme. If the fingerprint information is cracked by adversaries, it will pose a security threat to the user. So this scheme cannot protect the user's privacy very well. The scheme [20] proposes a solution that enables a user to authenticate with a remote server through a password and fingerprint. However, this scheme requires a secure channel between the client and the server, which is difficult to deploy in the actual scenario. Moreover, the mobile device is also required to be trusted. It means that if the mobile device is lost, the ciphertext and the fingerprint parameters stored on the phone can be cracked by attackers.

The password authentication method based on mobile phone assistance has received extensive attention and has been widely studied. However, these schemes have security risks in terms of user's privacy: the mobile phone has to store the user's privacy or credentials during the authentication process and the mobile phone must be trusted. Otherwise the user's privacy stored in the mobile phone can be cracked forcedly by the adversary. In March 2016, FBI announced that

it can crack the IOS to obtain the suspect's mobile phone terminal information [21]. This means that there is no mature technology that can fully guarantee the user's privacy stored in the mobile devices at present. So it is risky to store the user's secret information on the mobile phone. This risk will affect the security of the authentication scheme based on mobile phone assistance extensively.

Although authentication schemes assisted by mobile phone are popular, there are a lot of defects and hidden danger in this kind of authentication schemes. At present, most authentication schemes with mobile phone assistance have to store the user's private information on the mobile phone to help authentication. Moreover, in most schemes, the mobile phone is required to be fully trusted. However, in March 2016, FBI announced that it can crack the IOS system to obtain the suspect's mobile phone terminal information. This means that there is no mature technology that can fully guarantee the user's privacy stored in the mobile devices at present. So it is risky to store the user's secret information on the mobile phone. HCR (Hidden Credential Retrieval from a Reusable Password) [22] is a protocol that can be used by the user to store his information on an unreliable remote server. When the user wants to obtain his information, he can get his own encrypted information by a preset password and do not have to disclose the password to the server. However, this protocol needs to set up a secure channel to transmit the private key during the registration phase. Meanwhile, the private key has been saved on the server for a long time, which is an insecure operation. Existing three party authentication schemes participated with mobile phone, computer, and server often require the user's encrypted information saved on the mobile phone; meanwhile, the mobile phone must be reliable. What is more, the scheme assumes a secure channel to transmit secret information. This assumption and deployment are too difficult to be implemented in an actual situation, which greatly reduces the availability and security of the authentication scheme.

To solve these difficult problems, we propose a fingerprint protected password authentication (FPPA). Our new scheme has eliminated the complete credibility of the phone and the security channel between the phone and the server. When the user logs in he uses his mobile phone to generate the private key which is used to decrypt the ciphertext generated during the registration phase. The user needs to enter his password and fingerprint at the private key generation process. When the computer interacts with the mobile phone, the user's password will be blind. So the password can be protected from adversaries' attacks. Our scheme does not need to store any user's secret information on the mobile phone. Even if the mobile phone is stolen by adversaries, the private information in it will not be leaked. Security analysis proves that this scheme can protect the user's password against dictionary attacks and session hijacking attacks even without the assumption of secure storage and a secure channel between phone and server. Experimental results show that our scheme is essentially the same as the current schemes in terms of performance and has a significant advantage in the storage of mobile terminal over other schemes. The main contributions of our work can be summarized as follows:

(I) In order to ensure the security of the user's secret information in the IIOT, a fingerprint protected password authentication protocol which has no need to store the secret parameter in the mobile phone is proposed.

(II) Our scheme can resist the opponent's dictionary attacks, replay attacks, and phishing attack.

(III) Our scheme has good performance and strong practicability.

The rest of the paper is organized as follows. Section 2 introduces the preliminary knowledge of the paper. Section 3 introduces the model of the system model and the adversary model respectively. Section 4 introduces the overview of the FPPA scheme. Section 5 describes the security analysis and proof. Performance analysis and evaluation are presented in Section 6. Finally, Section 7 concludes.

2. Preliminary

2.1. Blind Signature. Blind signature scheme is a basic cryptographic primitive to guarantee the anonymity of participants. A blind signature scheme consists of two entities: a message sender and a signer. It allows the sender to obtain the signature of a given message without revealing any information about the message and the corresponding signature. The basic principle of blind signature is the application of two commutative algorithms. One algorithm is to conceal information, called blind transformation, and the other is the signature algorithm.

The characteristics of blind signature are as follows:

(I) The content of the message is blind to the signer.

(II) The signer cannot associate the signed message with the actual message signed. Even if he saves all the documents he signed, he cannot distinguish the real content of the documents he signed.

2.2. HKDF (Halting Key Derivation Functions). HKDF [23] is a protocol that can generate strong security key through a general weak password. It can choose the cost of the calculation, which is applied to the password-based encryption system.

HKDF includes two algorithms: *HKDF.prepare*() and *HKDF.extract*().

(I) $HKDF.prepare(w, t, r) \longrightarrow y, v$. *HKDF.prepare*() takes as input a password w, a random string r, and a parameter t of cycle calculations and returns a random token y and its ciphertext v.

(II) $HKDF.extract(w, v) \longrightarrow y$. *HKDF.extract*() takes as input a password w and a ciphertext v and either returns a token y or fails to halt in polynomial time. If the password w is not correct, this algorithm will always be a loop operation without output feedback 0.

In this paper, we use a *HKDF* algorithm to generate a strong key using a private key, and user or system can establish a suitable parameter value to select the corresponding calculation cost.

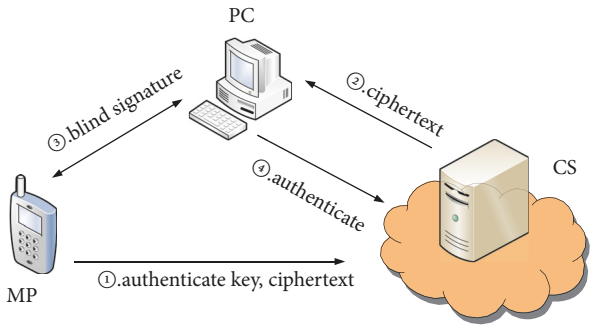

Figure 1: System architecture of our scheme.

3. System and Adversary Model

3.1. System Model. The system model of the scheme is shown in Figure 1. It mainly includes the following three entities: mobile phone (MP), personal computer (PC), and cloud server (CS).

MP is in charge of user's authentication key and generation of the ciphertext during the registration phase. It interacts with the PC in the authentication phase, so that the user's blind password and fingerprint parameters are processed during this phase. In addition to the basic functions of the mobile phone, it should be able to register the user's fingerprint data and connect to the PC via Bluetooth, USB, or Wi-Fi, in order to communicate with PC.

PC is responsible for the password input and interacts with mobile phone to generate session key when the user logs on. PC should ensure that it is connected to the Internet and can communicate with the CS and mobile phones.

CS stores users' authentication key and their ciphertext at the end of the registration phase and provides users with their corresponding ciphertext information. Meanwhile, it verifies user's identity in the login phase. Accordingly, the CS should be able to connect to the Internet and has a powerful database to store user's information.

The registration phase includes step ①, and the login phase includes steps ②, ③, and ④.

3.2. The Adversary Model. The adversary model is defined as follows:

(I) PC is distrusted. In the login phase, if the user's PC is infected by malwares, the keyboard may record the user's password. If the user accesses a phishing site, the adversary may also record the user's password.

(II) User uses mobile phone during the registration and login phase, while the mobile phone is prone to be lost or be implanted with malwares. If the phone was infected by malwares, the adversary may impersonate as the user to communicate with the PC.

(III) The CS authenticates the user and provides application services. In the registration phase, the CS is trusted. After that, the CS may suffer from replay attack or dictionary attacks. In addition, the CS may also be curious about user information.

4. The FPPA Scheme

4.1. Design Rationale of the FPPA Scheme. Our FPPA protocol allows a user register and authenticate with online services using his password and mobile phone securely. During the authentication protocol, the mobile phone may be lost or compromised by malwares, so the information stored on the mobile phone may be exposed by adversaries. At the same time, when the user logs on, the mobile phone interacts with the PC. If the mobile phone gets the user's password, it may also be stolen by the adversary. Therefore, it is required that authentication information and encrypted data should not be stored in the mobile phone. To avoid the case that the password is stolen by adversaries during the authentication phase, it is required that the PC encrypts the password after the user enters his password.

To this end, we improve the existing HCR protocol, which is deployed on the PC and mobile phone, such that user can interact with the distrusted mobile phone. In the HCR protocol, the private key needs to be transmitted in a secure channel and stored safely for a long time. Our scheme replaces the private key parameter with the user's fingerprint so that our scheme does not require the secure channel, or the long-term storage.

4.2. The Detailed Process of the FPPA Scheme. Our scheme includes a mobile phone, a PC, and a CS during the registration phase and the login phase. The user enters his password and fingerprint on the phone and PC to register and log on. The CS is responsible for the registration and authentication when the user wants to access the CS.

The system initialization definition is as follows:

p: a prime number;

name: the user name;

pwd: the user's password;

G: a cyclic abelian group of order p;

F: a multiplicative domain of order p;

e: the user's fingerprint;

s: the user's private key;

y: the user's authentication key;

v: the ciphertext of the authentication information;

$Hash() : \{0,1\}^* \longrightarrow G$: a cryptographic hash function, which is to be viewed as a random oracle;

$d \in F_p^x$: a random number generated by the system;

f: a hash function;

HKDF: a key derivation function, which consists of two functions: *HKDF.prepare*() and *HKDF.extract*().

Generally, in the previous mobile phone based authentication scheme, the mobile phone stores user's authentication information. The user's password is encrypted and stored on the mobile phone, or the user's secret key is encrypted with his password and stored on the mobile phone. If the phone is compromised by the adversary, the adversary is likely to get the user's password or authentication information.

In our FPPA scheme, the mobile phone does not store the user's authentication information. The authentication key is regenerated by himself when he wants to log on. Even if the device is stolen, it will not pose threat to the user's password,

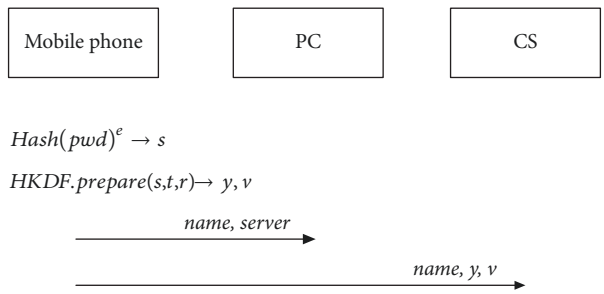

Figure 2: Registration phase.

fingerprint, or encrypted information. Moreover, when the user sends the name to the server in the registration phase, he/she can also choose to send a random identity ID, so that he/she can preserve his/her anonymity. Even if several CSs are in collusion with each other, the user cannot be attacked.

Registration and login phase are as follows:

4.2.1. Registration Phase. In this phase, before the user accesses to the service provided by the CS, he/she is required to register to be a legal user. The user can register through the mobile phone, after the completion of registration phase, the user name *name* and service information are sent to the PC side.

The process is detailed as follows, as is shown in Figure 2.

(I) Mobile phone generates the user's private key s according to the user's password *pwd* and fingerprint parameter *e*. The user's private key *s* is generated as follows:

$$Hash(pwd)^e \longrightarrow s. \quad (1)$$

(II) The mobile phone side generates the user's authentication key *y* and the ciphertext of authentication key *v* based on the HKDF function and the private key *s*. The operation is as follows:

$$HKDF.prepare(s,t,r) \longrightarrow y, v, \quad (2)$$

where *r* is a random string generated by the system; *t* is the number of loop operations. Note that *t* can be selected according to the security requirements of the user. If the value of *t* is larger, the computation would be more time consuming. So the generated key is more secure.

(III) The mobile phone sends the authentication key *y*, the ciphertext *v*, and the user name *name* to the CS. Meanwhile, the mobile phone sends the user name *name* and CS information *server* to the PC. The user just remembers the user name *name* and password *pwd*, and the mobile phone does not store any secret information of the user.

4.2.2. Login Phase. When the user wants to access the CS, he/she needs to send a login request to the CS. The user enters his own password and fingerprint data to generate a private key and then decrypts the ciphertext to obtain the authentication key. The server determines whether to accept the logon request or not after the server computes and compares the authentication information. The specific operation is as follows, as is shown in Figure 3

(I) The PC sends *name* to the CS, and then the CS retrieves the corresponding information according to *name*. After that, the CS sends the corresponding ciphertext *v* and certified random number *chal* to the PC.

(II) The user enters his password on the PC. The system generates a blind parameter *d* and then calculates the blinded password μ to the mobile phone.

$$Hash(pwd)^d \longrightarrow \mu \quad (3)$$

(III) The user inputs his fingerprint on the mobile phone. The mobile phone gets the fingerprint parameter ε and then obtains β by signing the blinded password μ. The mobile phone sends β to the PC.

$$\mu^{\varepsilon} \longrightarrow \beta \quad (4)$$

(IV)The PC recovers the private key *s* by the following calculation.

$$\beta^{1/d} \longrightarrow s \quad (5)$$

(V)The PC uses the private key *s* and HKDF algorithm to decrypt the ciphertext *v* and then obtains the authentication key *v*.

$$HKDF.extract(s,v) \longrightarrow y \quad (6)$$

(VI)The PC obtains the response by computing $f(y, chal) \longrightarrow response$ and sends the response *response* to the CS.

(VII)Finally, the CS computes $response' \longleftarrow f(y, chal)$ and compares it with the response received. This logon is successful if these two parameters are identical.

5. Security Analysis and Proof

5.1. FPPA Protocol. Our FPPA protocol has three types of entities: the PC client who wants to use a password and fingerprint to access services, the CS who registers and authenticates clients, and mobile phone who assists PC client to complete registering and logging.

Our FPPA protocol consists of following algorithms:

UserGen. This algorithm is run by the user to generate a user name *name* and an l-bit password *pwd*.

Register. The user registers with the server by inputting his password and fingerprint on the mobile phone. In the end, the mobile phone outputs an authenticated key *y* and ciphertext of the authenticated key *v*. The mobile phone sends (*name*, *y*, *v*) to the CS and sends (*name*, *server*) to the PC.

Store. The PC stores the user name *name*. The mobile phone does not store user's any information. The user just needs to remember his name and password.

Retrieve. The PC client uses its user name *name* to retrieve its ciphertext *v* from the server. The server sends the ciphertext *v* and challenge *chal* to the PC client.

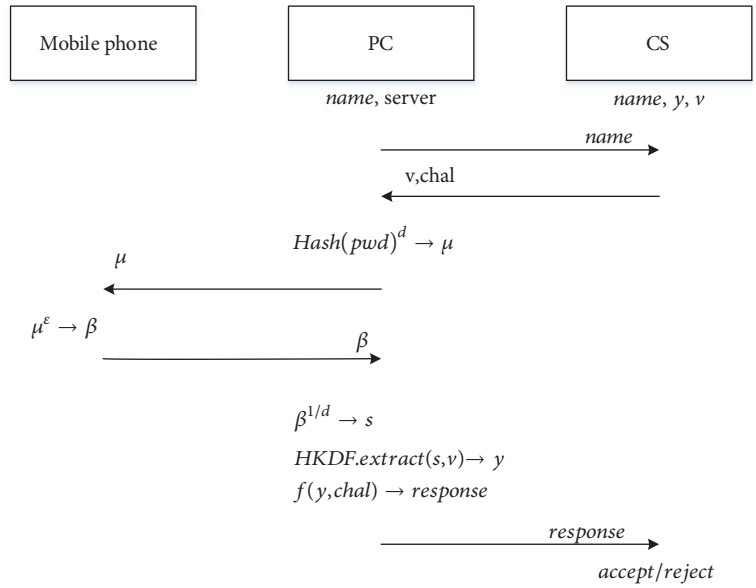

Figure 3: Login phase.

PreAuth. The user inputs his password *pwd* in the PC side and inputs his fingerprint *e* on the mobile phone. The PC client interacts with the mobile phone using the blind signature. At the end of the blind signature, the PC client generates the private key *s* and decrypts the ciphertext *v* using *s*. The PC client obtains the authenticated key *y* after decrypting the ciphertext.

Authenticate. The user uses his authenticated key *y* and change *chal* to prove to the server that he owns the corresponding account credential. Finally, the CS outputs accept or reject.

5.2. Security Game. Our FPPA scheme assumes that the user has a mobile phone and PC client, so that the user can enter his own password and fingerprint on these devices. However, mobile phones may be lost. Therefore, malicious adversaries may use the user's data stored on the mobile phone to access the server, and even decrypt the ciphertext brutally. What is more, the PC client and the server are also easily attacked by adversaries [24, 25]. We divide the potential attacks into two types: Insider attack (Game One) and Outsider attack (Game Two). Let x be a security parameter, such that all hash functions have at least l≥2x bits of output. Let $|D| << 2^x$ be the length of the password dictionary. k is the length of the user's password. $k \in N$. Let neg(k) be a negligible function of k if ∃K finite:∀k>K: $neg(k) < k^{-c}$[26–28].

Game One (Outsider Attack). In this game, the adversary can control the PC side. He can perform the Store step with the mobile phone and PC. He can also perform the *PreAuth* step with the server by acting as a malicious PC. The adversary may also control the PC side and then launch the Retrieve request. We assume the times of requests are q. The adversary can control the mobile phone side and perform Retrieve step with the PC side by simulating the mobile phone side. In the algorithm of our scheme, mobile phone can only receive data from the PC side but cannot initiate requests to the PC side. If the adversary got the user's password through the game or guessed the private key s without the password, we consider the adversary win [29, 30].

Definition 1. We assume that the probability of adversary winning the Game One is P1. If P1≤max[q/|D|+neg(k), 1/2|s|], we consider that FPPA is in accord with the security based on Game One.

Game Two (Insider Attack). In this game, the adversary can control the server side. He can launch an offline dictionary attack on the server to get the ciphertext of the user's password. We assume that the adversary can initiate *T* times offline dictionary attacks against *HKDF.extract*. If the adversary got the user's password through the game, we consider the adversary wins.

Definition 2. We assume that the probability of the adversary winning the Game Two is *P2*. If $p2 \le q/|D|+2T/|D|t+neg(k)$, we consider that FPPA is in accord with the security based on Game Two.

Definition 3. If FPPA is in accord with the security based on Game One and Game Two, we consider the FPPA is security.

5.3. Security of FPPA

Theorem. *FPPA is in accord with the security based on Definition 3.*

Prove. (I) FPPA can provide security based on Game One.

Game One. (1)The adversary simulates the PC side and performs *Store* with mobile phone to get the user's name *name*. The adversary performs *PreAuth* with server side and retrieves user secret key ciphertext v and challenge *chal* according to the user's name. According to v is generated by the algorithm of *HKDF* [23], if someone inputs wrong password to decrypt the ciphertext, the *HKDF* would compute circularly with no output. So the adversary cannot get user's password through v. Consequently, *P1* is a small value that can be ignored.

(2)The adversary simulates the PC side, generates a password *pwd0*, and performs *Retrieve* with mobile phone. This interaction process is similar to the interaction process in the algorithm of *HCR* and just replaces the signature private key in *HCR* into fingerprint parameters. The probability of guessing password in this game is the same as that in HCR [22]. So the probability of the adversary guessing the password: $P[\text{guess password}] \leq q/|D| + neg(k)$. The probability of the adversary winning this game: $P1 \leq q/|D| + neg(k)$.

(3)The adversary controls the PC side and guesses the private key s without password. The probability of the adversary guessing private key: $P[\text{guess s}] \leq 1/2^{|s|}$. The probability of the adversary winning this game: $P1 \leq 1/2^{|s|}$.

(4)The adversary performs *Retrieve* with PC side by simulating mobile phone. The adversary can get the blinded password of the user. So the probability of the adversary guessing the password is the same as the analysis of (b). The probability of the adversary guessing the password: $P[\text{get the password}] \leq q/|D| + neg(k)$.The probability of the adversary winning this game: $P1 \leq q/|D| + neg(k)$.

In summary, $P1 \leq \max[q/|D| + neg(k), 1/2^{|s|}]$. So FPPA can provide the security based on Game One.

(II) FPPA can provide security based on Game Two.

Game Two. Malicious server launches offline dictionary attacks on the ciphertext stored on itself. It generates a private key s' and tries to decrypt ciphertext: *HKDF.extract (s',v)*. According to the security of *HKDF*[23]:

$P[getthepassword] \leq T/|D|t + neg(k)$ .t is the parameter of cyclic operation times set up in *HKDF.prepare*.

So the probability of the adversary getting the user's password: $P2 \leq q/|D| + 2T/|D|t + neg(k)$.

Consequently, FPPA can provide the security based on Game Two.

In summary, FPPA can provide security based on Game One and Game Two, so FPPA is a secure protocol.

6. Performance Analysis and Evaluation

6.1. Test Plan and Scenario. We rented Ali CS to authenticate with the user in order to make the test scenario more close to the actual one. Ali CS deployed in Qingdao and the distance between Qingdao and Xi'an is 1058.6km.

As shown in the Figure 4, 1 describes that the mobile phone computes and generates authenticate key y and authenticate key ciphertext v. And then the mobile phone sends v to the CS in the end of registration phase. 2, 3, and 4 describe that the mobile phone and the PC compute and generate authenticate key y to authenticate with CS in login phase.

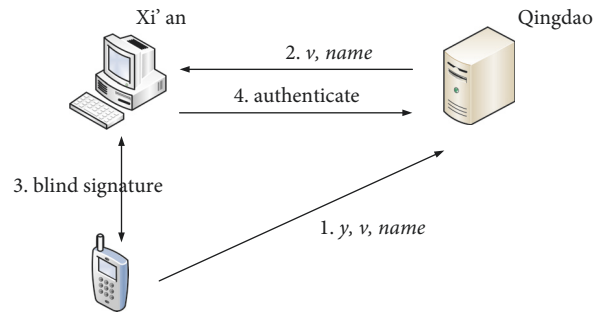

Figure 4: The overall test scenario.

Android device, a PC and CS are used to simulate our scheme. We evaluate the usability of our scheme from the aspects of time and storage. We test the user's registration and login time in four different scenarios. In terms of storage capacity, we mainly analyze and evaluate the storage capacity of the Android device. The experimental devices of our scheme are a PC, a CS, and an Android device. The specific parameters of the experimental devices are as follows:

We use Ali CS located in Qingdao as the CS in our scheme. RAM of the CS is 1GB. It has a single-core processor. The operating system of it is windows Server 2008.

We use HP Compad dx7408 MT DT PC as the computer in our scheme. It has a dual-core processor. RAM of the computer is 2GB. The operating system of it is windows 7(32 bits).

We use Bluestacks (Android 4.1.2-API Level 16, CPU ARM (armeabi), RAM 512, VM Heap 16, Internal Storage 200MiB) and Android phone as the mobile phone in our scheme.

We tested our scheme in 4 cases.

Case 1. The user registers and logs in, and then enters his password and fingerprint. We test the time spent on registration phase and login phase and then test several sets of data to find the average time for one user spent on registration and login phase. The Android device used in this scenario is the Android simulator.

Case 2. The length and complexity of passwords are different for different users. In order to analyze the influence of different passwords on our scheme, we select 20 groups of users. We test the time of these users spent on registration phase and login phase. The Android device used in this scenario is the Android simulator.

Case 3. Due to the different system settings, different length of fingerprint parameters is generated in this case. We test the time spent on registration phase and login phase while the length of the fingerprint is 128b, 256b, and 512b. The Android device used in this scenario is the Android simulator.

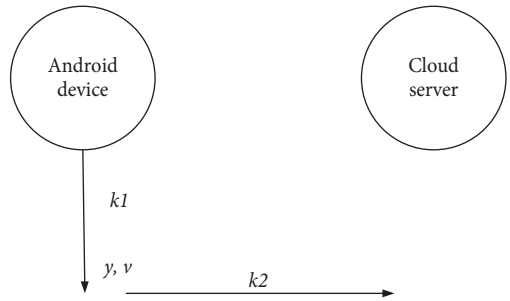

Figure 5: Registration phase.

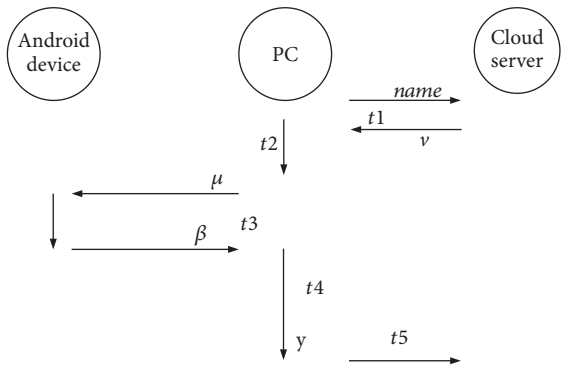

Figure 6: Login phase.

Case 4. The user uses different Android devices (Android emulator, Meizu mobile phone, and HTC mobile phone) for registering and logging in.

We test the time spent on registration phase and login phase. We divide the total time of registration phase K into two parts: $k1$ and $k2$ in this experiment. As shown in Figure 5, $k1$ms is the time spent on the registration phase starting from the user's registration on the Android device to the end of the generation of authenticated key ciphertext. $k2$ms is the time spent on sending the authenticated key to the CS.

The total time T ms of the login phase is divided into five parts. As shown in Figure 6, t1 is the time spent on requesting the corresponding authentication key ciphertext from the user's PC. t2+t4 is the time spent on blinding the user's password and generating the authentication key. t3 is the time spent on signing the blinded password with fingerprint data on the Android device. t5 is the time spent on sending the user authentication information to the CS.

6.2. Experiment Data

6.2.1. Experiment Data of Case 1

(I) Registration Phase. In this case, the user enters the user name, password, and fingerprint on the Android device for registration. The length of user's fingerprint is 256b. As shown in Table 1, we measured 10 groups of data. We can calculate that the average registration time is 216.9ms through the following data.

Table 1: Registration phase time of Case 1.

Test Group	Registration Phase Time/ms		
	k1	k2	K
1	197	19	216
2	200	18	218
3	199	18	217
4	210	17	227
5	199	15	214
6	198	18	216
7	196	17	213
8	190	19	209
9	213	17	220
10	199	20	219

Table 2: Login phase time of Case 1.

Test Group	Login Phase Time/ms				
	t1	*t3*	*t2+t4*	*t5*	*T*
1	39	90	36	19	184
2	45	98	39	18	200
3	38	94	41	18	191
4	38	92	38	16	184
5	37	110	40	17	204
6	39	98	39	19	195
7	40	96	35	17	188
8	41	97	41	19	198
9	39	92	40	18	189
10	38	99	38	17	192

(II) Login Registration. When the user logs in, he enters his password and fingerprint on the PC and Android emulator. We measured the time spent on each stage and then measured 10 groups of data showing in Table 2. We can calculate the average login time is 192.5 ms through these following data.

6.2.2. Experiment Data of Case 2

(I) Registration Phase. In this case, we test 20 groups of different users. The user inputs his name, password, and fingerprint on Android emulator for registration. We measure the time (k1, k2) spent during the registration phase. The complexity and length of the user name and password are different. The length of these users' fingerprint is 256 bits. The measured result is shown in Table 3.

(II) Login Phase. Users enter their password and fingerprint on the PC and Android devices. We measure the time spent on each stage. 20 groups of data are shown in Table 4.

6.2.3. Experiment Data of Case 3

(I) Registration Phase. The user inputs his password on Android device, and then the Android device generates three kinds of fingerprint data with different length (512b, 256b,

Table 3: Registration phase time of Case 2.

User	Registration Phase Time/ms		
	k1	*k2*	*K*
User1	222	15	237
User2	195	17	212
User3	271	17	288
User4	236	19	255
User5	243	20	263
User6	190	17	207
User7	220	18	238
User8	214	16	230
User9	195	21	216
User10	239	18	257
User11	197	16	213
User12	220	19	239
User13	230	18	248
User14	220	17	237
User15	200	18	218
User16	261	19	280
User17	199	16	215
User18	215	18	233
User19	250	19	269
User20	234	17	251

Table 4: Login phase time of Case 2.

User	Login Phase Time/ms				
	t1	*t3*	*t2+t4*	*t5*	*T*
User1	45	88	36	17	186
User2	37	94	38	17	186
User3	37	93	41	15	186
User4	38	63	35	18	154
User5	37	90	41	20	188
User6	39	111	39	19	208
User7	37	93	42	17	189
User8	40	98	36	16	190
User9	39	90	38	21	188
User10	37	92	38	19	186
User11	38	94	42	17	191
User12	40	80	35	18	173
User13	37	98	36	19	190
User14	39	91	44	18	192
User15	36	89	36	19	180
User16	38	92	40	17	184
User17	37	100	39	20	196
User18	38	93	37	19	187
User19	39	92	40	17	188
User20	37	94	39	18	188

and 128b). We measure the time (k1, k2) spent during the registration phase. The measured result is shown in Table 5.

Table 5: Registration phase time of Case 3.

Length Of Fingerprint/b	Registration Phase Time/ms		
	k1	*k2*	*K*
512	210	19	229
256	205	18	223
128	192	18	220

Table 6: Login phase time of Case 3.

Length Of Fingerprint/b	Login Phase Time/ms				
	t1	*t3*	*t2+t4*	*t5*	*T*
512	39	100	39	18	196
256	39	98	38	18	193
128	38	95	38	17	186

(II) Login Phase. The user inputs his password and fingerprint on PC and Android devices. We test the time spent during the login phase. The measured result is shown in Table 6.

6.2.4. Experiment Data of Case 4

(I) Registration Phase. In this case, there are three different Android devices (Android simulator, Meizu phone, and HTC phone). The user enters his password and fingerprint on different Android devices. We measure the time spent during this phase (*k1, k2*).

(II) Login Phase. The user enters his password and fingerprint on PC and different Android devices. We test the time spent during the login phase. The measured result is shown in Table 8.

6.3. Performance Analysis. We can calculate the average time the user registered on the Android simulator is 216.9ms according to the Table 1. The average time required for the user to log in using the Android emulator and PC is 192.5ms.

As shown in Table 3, when different users enter a different password to register, the difference between the longest and the shortest time is 81ms which is less than the human reaction time. Users cannot feel the difference in the length or complexity of passwords. Therefore, the length and complexity of the password have little impact on the user's registration time. The comparison of the time spent by different users in the registration phase is shown in Figure 7.

As shown in Table 4, when different users enter their password to login, the difference between the longest and the shortest time is 54ms, which is less than the human reaction time. Therefore, the length and complexity of the password have little impact on the user's login time. The comparison of the time spent during the login phase is shown in Figure 8.

When the length of the fingerprint is different, the maximum difference of the time among users during the registration phase is 9ms. The maximum difference of the time spent in the login phase is 10ms which has little effect on users. When the length of the fingerprint is 128b, 256b,

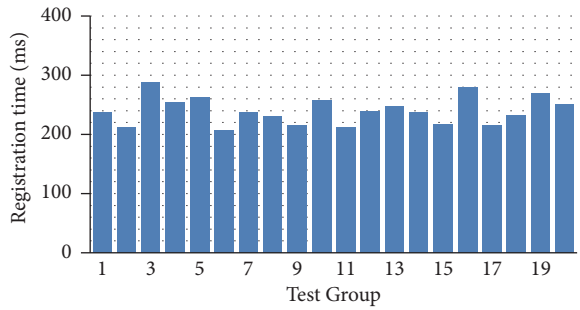

Figure 7: Registration phase experimental data of Case 2.

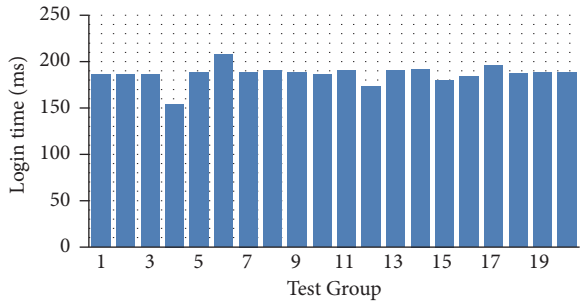

Figure 8: Login phase experimental data of Case 2.

Table 7: Registration phase time of Case 4.

Device	Registration Phase Time/ms		
	k1	*k2*	*K*
Bluestacks	200	17	217
MEIZU	18	19	37
HTC	19	21	40

Table 8: Login phase time of Case 4.

Device	Login Phase Time/ms				
	t1	*t3*	*t2+t4*	*t5*	*T*
Bluestacks	39	96	39	18	192
MEIZU	40	56	35	21	152
HTC	38	57	36	20	151

and 512b, the comparison of the time spent in the registration phase and the login phase is shown in Figure 9.

In Case 4, the user enters the same password and fingerprint on different Android devices. As shown in Tables 7 and 8, the total time spent on registering and logging in on mobile devices is less than the total time spent on the Android simulator. There is no difference in their transmission time. The main difference is the calculation time on Android devices. In the existing Mobile SPA scheme, the total time (26ms) spent in the registration phase is the transmission time and the computation time on the mobile terminal. The time spent on the two mobile phones in this paper is 37ms and 40ms. The difference of the time between them is small, so it has little effect on the performance of the scheme. In the SPA Mobile scheme, the time (38ms) spent in the login phase

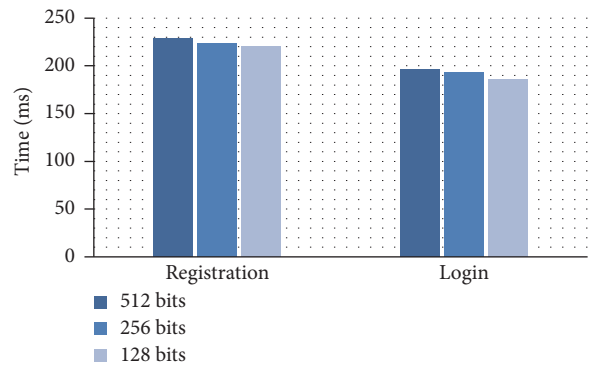

Figure 9: Comparison of experimental data in two phases of Case 3.

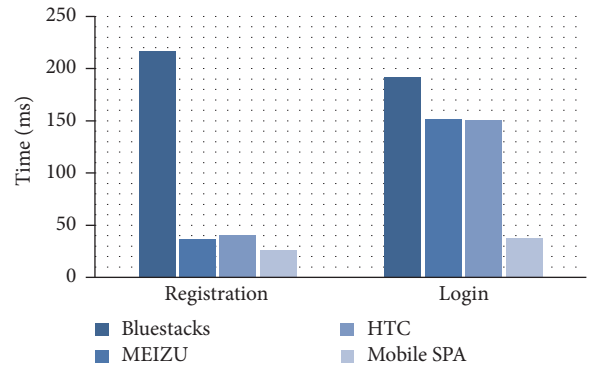

Figure 10: Comparison of experimental data in two phases of Case 4.

is the computation time on the server side, computing time on the mobile phone, and the transmission time. The total time of our scheme is 152ms and 151ms. Compared with the scheme of Mobile SPA, the time difference of our scheme is larger. However, our scheme has blind signature operation in the login phase and regeneration of the key operation and so on which greatly improve the security of the system. The time Users' using different Android device in registration and login is shown in Figure 10.

We can see from above test scenarios that the running time of our scheme during the registration phase is k1; the running time of our scheme in the login phase is t3+t2+t4; the total running time of our scheme is k1+t3+t2+t4 called w. The propagation delay time between the computer and the mobile phone is brief, so we consider t3 as the running time in the Android device. We can calculate the total running time w according to the Case 4. The comparison diagram of the time running in the distinct Android device is shown in Figure 11. We can see from Figure 11 that our scheme running in the Bluestacks costs the longest time. That is because there are some gaps in processor and set parameters between Bluestacks and Android mobile phone which influences the efficiency of our scheme.

In the storage capacity of mobile phones, the Mobile SPA scheme requires storing MAC key on the phone to authenticate with the CS. A server corresponds to a MAC key,

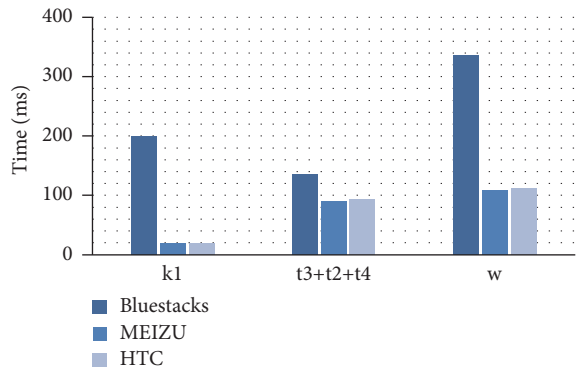

Figure 11: Comparison of the running time of different devices.

while the length of a MAC key is 128 bytes. In our scheme, the mobile device only needs to store the user's fingerprint data. The maximum length of fingerprint data is 512 bits (64 bytes). If the user registers 2000 servers, in the Mobile SPA scheme, the user requires 250kb storage capacity while the user requires 0.0625kb storage capacity in our FPPA scheme. Meanwhile, our scheme does not need to store data in the mobile phone for a long time.

In conclusion, although our scheme has a significant increase in the login phase, the user has little effect on the use in terms of user experience. Meanwhile, our scheme adds the blind signature operation and regeneration of the key operation and so on which greatly improve the security of the system. What is more, our scheme only stores the user's fingerprint in the mobile phone temporary. Because only a small amount information is stored in our scheme, the storage pressure of the mobile terminal is reduced. Therefore, our scheme has a high application value.

7. Summary

Passwords are widely deployed to secure users' access to cloud services in IIOT. However, passwords are prone to dictionary attack and phishing. Therefore, protecting the user's password when authenticating the user's identity has become a widespread concern. In this paper, we use the mobile phone to assist user's authentication. Specifically, we use both password and fingerprint to generate a secret key and then we use the secret key and HKDF function to generate the authentication key. Due to the characteristic of HKDF function, any adversary cannot get the key or password or decrypt the ciphertext through exhausting enumerating attacks. There is no user's secret information on the mobile phone, so even if the mobile phone is lost, it will not pose a security risk to the user's account. Finally, we prove the security of the scheme and evaluate the performance of our scheme. Results show that our scheme can resist dictionary attacks and attacks against the user's mobile phone. Meanwhile, there is no significant difference between our scheme and the existing three party authentication schemes, in terms of performance. In addition, our scheme reduces the storage pressure on the mobile phone. Overall, our scheme has high practicality.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work is supported by the National Key R&D Program of China (Grant no. 2017YFB0801805), the General Program of National Natural Science Foundation of China (Grant no. 61672415), the 111 Project (Grant no. B16037), and the open fund project of Science and Technology on Communication Networks Laboratory (Grant no. SXX18641X024).

References

[1] H. Xiong, H. Zhang, and J. Sun, "Attribute-based privacy-preserving data sharing for dynamic groups in cloud computing," *IEEE Systems Journal*, pp. 1–22, 2018.

[2] C.-C. Lee, T.-H. Lin, and C.-S. Tsai, "A new authenticated group key agreement in a mobile environment," *Annals of Telecommunications-Annales des Télécommunications*, vol. 64, no. 11-12, pp. 735–744, 2009.

[3] J. Shen, T. Zhou, X. Chen, J. Li, and W. Susilo, "Anonymous and Traceable Group Data Sharing in Cloud Computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 4, pp. 912–925, 2018.

[4] J. Xiong, Y. Zhang, and L. Lin, "ms-PoSW: A multi-server aided proof of shared ownership scheme for secure deduplication in cloud," *Concurrency & Computation Practice & Experience*, vol. 5, Article ID e4252, 2017.

[5] M. Zhang, Y. Yao, Y. Jiang, B. Li, and C. Tang, "Accountable mobile E-commerce scheme in intelligent cloud system transactions," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 6, pp. 1889–1899, 2018.

[6] D. Wu, F. Zhang, H. Wang, and R. Wang, "Security-oriented opportunistic data forwarding in Mobile Social Networks," *Future Generation Computer Systems*, vol. 87, pp. 803–815, 2018.

[7] D. Wu, S. Si, S. Wu, and R. Wang, "Dynamic trust relationships aware data privacy protection in mobile crowd-sensing," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2958–2970, 2018.

[8] D. Wu, J. Yan, H. Wang, D. Wu, and R. Wang, "Social attribute aware incentive mechanism for device-to-device video distribution," *IEEE Transactions on Multimedia*, vol. 19, no. 8, pp. 1908–1920, 2017.

[9] N. Carlson, "Mark Zuckerberg broke into a facebook user's private email account," 2010, http://www.businessinsider.com.

[10] Q. Jiang, M. K. Khan, X. Lu, J. Ma, and D. He, "A privacy preserving three-factor authentication protocol for e-Health clouds," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3826–3849, 2016.

[11] C. Li, C. Lee, and C. Weng, "An extended chaotic maps based user authentication and privacy preserving scheme against DoS attacks in pervasive and ubiquitous computing environments," *Nonlinear Dynamics*, vol. 74, no. 4, pp. 1133–1143, 2013.

[12] Q. Jiang, J. Ma, C. Yang, X. Ma, J. Shen, and S. A. Chaudhry, "Efficient end-to-end authentication protocol for wearable health monitoring systems," *Computers & Electrical Engineering*, vol. 63, pp. 182–195, 2017.

[13] M. Wu, S. Garfinkel, and R. Miller, "Secure web authentication with mobile phones," in *Proceedings of the DIMACS workshop on usable privacy and security software*, 2004.

[14] D. Thanh, I. Jorstad, and T. Jonvik, "Strong authentication with mobile phone as security token," in *Proceedings of the IEEE 6th International Conference on Mobile Adhoc and Sensor Systems*, pp. 777–782, IEEE, Macau, China, 2009.

[15] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-is-believing: using camera phones for human-verifiable authentication," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 110–124, IEEE, May 2005.

[16] M. Mannan and P. C. van Oorschot, "Using a personal device to strengthen password authentication from an untrusted computer," in *Financial Cryptography and Data Security*, pp. 88–103, Springer Berlin Heidelberg, Berlin, Germany, 2007.

[17] C.-L. Chen, C.-C. Lee, and C.-Y. Hsu, "Mobile device integration of a fingerprint biometric remote authentication scheme," *International Journal of Communication Systems*, vol. 25, no. 5, pp. 585–597, 2012.

[18] T. Acar, M. Belenkiy, and A. Küpçü, "Single password authentication," *Computer Networks*, vol. 57, no. 13, pp. 2597–2614, 2013.

[19] Q. Su, J. Tian, and X. Chen, "A fingerprint authentication system based on mobile phone," in *Proceedings of the Audio and Video-Based Biometric Person Authentication*, pp. 151–159, Springer Berlin Heidelberg, New York, NY, USA, 2005.

[20] G. Starnberger, L. Froihofer, and K. M. Goeschka, "QR-TAN: Secure mobile transaction authentication," in *Proceedings of the International Conference on Availability, Reliability and Security (ARES '09)*, pp. 578–583, IEEE, Fukuoka, Japan, March 2009.

[21] Z. Matt, "FBI has accessed san bernardino shooter's phone without apple's help," 2016, http://gadgets.ndtv.com/mobiles/news.

[22] X. Boyen, "Hidden credential retrieval from a reusable password," in *Proceedings of the 4th International Symposium on ACM Symposium on Information, Computer and Communications Security (ASIACCS '09)*, pp. 228–238, ACM, New York, NY, USA, March 2009.

[23] X. Boyen, "Halting password puzzles," in *Proceedings of the 16th USENIX Security Symposium on USENIX Security Symposium*, pp. 119–134, ACM, Berkeley, Calif, USA, 2007.

[24] F. Wei, P. Vijayakumar, Q. Jiang, and R. Zhang, "A mobile intelligent terminal based anonymous authenticated key exchange protocol for roaming service in global mobility networks," *IEEE Transactions on Sustainable Computing*, no. 99, pp. 2377–3782, 2018.

[25] Q. Jiang, Y. Qian, J. Ma, X. Ma, Q. Cheng, and F. Wei, "User centric three-factor authentication protocol for cloud-assisted wearable devices," *International Journal of Communication Systems*, vol. 32, pp. 1–20, 2019.

[26] F. Li, Y. Han, and C. Jin, "Cost-effective and anonymous access control for wireless body area networks," *IEEE Systems Journal*, vol. 12, no. 1, pp. 747–758, 2018.

[27] C.-C. Lee, M.-S. Hwang, and W.-P. Yang, "A new blind signature based on the discrete logarithm problem for untraceability," *Applied Mathematics and Computation*, vol. 164, no. 3, pp. 837–841, 2005.

[28] H. Xiong, Q. Mei, and Y. Zhao, "Efficient and provably secure certificateless parallel key-insulated signature without pairing for IIoT environments," *IEEE Systems Journal*, Article ID 2890126, pp. 1–11, 2018.

[29] S. Kumari, P. Chaudhary, C. Chen, and M. K. Khan, "Questioning key compromise attack on Ostad-Sharif et al.'s authentication and session key generation scheme for healthcare applications," *IEEE Access*, vol. 7, pp. 39717–39720, 2019.

[30] S. Kumari, X. Li, F. Wu, A. K. Das, K.-K. R. Choo, and J. Shen, "Design of a provably secure biometrics-based multi-cloud-server authentication scheme," *Future Generation Computer Systems*, vol. 68, pp. 320–330, 2017.