Computers, Materials & Continua CMC, vol.63, no.2, pp.663-674, 2020

Multi-Factor Password-Authenticated Key Exchange via Pythia PRF Service

Zengpeng Li[1], Jiuru Wang[2, *], Chang Choi[3] and Wenyin Zhang[2]

Abstract: Multi-factor authentication (MFA) was proposed by Pointcheval et al. [Pointcheval and Zimmer (2008)] to improve the security of single-factor (and two-factor) authentication. As the backbone of multi-factor authentication, biometric data are widely observed. Especially, how to keep the privacy of biometric at the password database without impairing efficiency is still an open question. Using the vulnerability of encryption (or hash) algorithms, the attacker can still launch offline brute-force attacks on encrypted (or hashed) biometric data. To address the potential risk of biometric disclosure at the password database, in this paper, we propose a novel efficient and secure MFA key exchange (later denoted as MFAKE) protocol leveraging the Pythia PRF service and password-to-random (or PTR) protocol. Armed with the PTR protocol, a master password pwd can be translated by the user into independent pseudorandom passwords (or rwd) for each user account with the help of device (e.g., smart phone). Meanwhile, using the Pythia PRF service, the password database can avoid leakage of the local user's password and biometric data. This is the first paper to achieve the password and biometric harden service simultaneously using the PTR protocol and Pythia PRF.

Keywords: Multi-factor authentication key exchange, biometric data, password-to-random, Pythia PRF.

1 Introduction

To steal the identity of the customer is easy in the computing world using a bit of social engineering. Once obtained the identity of the customer, the unauthorized attacker is massively more likely to access to the trustworthy information over the network. Thus, to avoid unauthorized persons to access sensitive information, various authentication methods can be adopted as the first line of defense against intruders, such as single-factor authentication (SFA) and two-factor authentication (TFA), where the password-based authentication key exchange is the well-recognized SFA scheme. Further, multi-factor authentication (MFA) was designed to overcome shortcomings and deficiencies of SFA and TFA while it provides a high level of security. However, other issues were introduced, such as **1).** Usability, customers have to manage an additional layer of

[1] College of Computer Science and Technology, Qingdao University, Qingdao, 266071, China.

[2] School of Information Science and Engineering, Linyi University, Linyi, 276005, China.

[3] IT Research Institute, Chosun University, Gwangju, 61452, South Korea.

[*] Corresponding Author: Jiuru Wang. Email: wangjiuru@lyu.edu.cn.

Received: 07 March 2019; Accepted: 20 July 2019.

security (e.g., biometric template and short message service (or SMS)) in addition to having to manage the password; and **2).** Cost, MFA brings potential cost increases for things like additional support, SMS Gateway or services, mobile app development, hardware and software tokens. To address all these issues simultaneously is an important and open topic. Pointcheval et al. [Pointcheval and Zimmer (2008)] proposed the first MFA key exchange (or MFAKE) scheme which adopts three types of authentication factors, namely: **1). knowledge**, a secret information or password (is something you know), **2). possession**, a physical secure device on you with a secret key (is something you have), and **3). Inherence**, a biometric (is something you are). To be specific, in multi-factor authentication, the secret information can be regarded as long-term (unchanging) passwords, and an unclonable and physical secure device with a secret key can be regarded as a short-term (changing) password, such as an electronic password token or sheet of paper. Nowadays, one-time password is one of the most methods to be adopted. In a nutshell, the workflow of Pointcheval et al. [Pointcheval and Zimmer (2008)] contains the following three stages; firstly, the user U and the server S accomplish a Diffie-Hellman key exchange instance using the password of user as the authentication information. Then to protect the biometric data of user, the server S adopts ElGamal encryption to encrypt each bit of the biometric template. Finally, the user can retrieve the fresh ciphertexts with the knowledge of biometric and one-time password.

Pointcheval et al. [Pointcheval and Zimmer (2008)] analyzed the security in BRP model where an adversary (merely) wants to impersonate the client, but they did not discuss what will happen if the attacker impersonates the server. Hao et al. [Hao and Clarke (2012)] proposed two immediate attacks on the scheme of Pointcheval et al. [Pointcheval and Zimmer (2008)], if the password has already been compromised by an attacker. One of the attacks is that an attacker can launch attack by stealing the victim' biometric, and another attack is to discover the one-time password using the Chinese Remainder theorem. Thus, the entire system can be broken when the password is compromised.

Further, in order to achieve multi-factor authentication, biometric template information (such as iris scans and fingerprint) is commonly used in the current technology. However, there is no literature to discuss how to securely store biometric by server. Hao et al. [Hao and Clarke (2012)] only gave an attack on how to use the stolen biometric and did not give an efficient method how to keep it securely. Obliviously, once the server is compromised, users must confront risks that biometric secrets may be leaked. The following problem attracts our attention.

How to keep the privacy of the biometric at the server side in MFA schemes?

To deal with the potential risk of biometric disclosure, in our framework, at the server side (or registration center), we leverage the Pythia PRF service to harden stored biometric hashes (or ciphertexts) and the password hashes against offline brute-force attacks. Meanwhile, at the client side, to enhance the security of MFA scheme, apart from we leverage the sensor of smartphone to fulfill the extraction of the biometric information, we use the smartphone as the accessibility tool to cooperate with the user and run the short authentication string (SAS) message authentication protocol and the password-to-random (PTR) protocol. Here, the SAS protocol can be used to check the received message integrity, and the PTR protocol can be used to harden is a password that allows a master

pwd can be translated into independent rwds with the help of device (e.g., smart phone, which stores a key) for each user account. The above mentioned is high-level description of our framework, the detailed constructions are presented in the following sections.

1.1 Related work

Single-factor authentication. Single-factor authenticated key exchange dominated the research of authenticated key exchange (AKE) protocols for a long time, which was first proposed by Bellovin et al. [Bellovin and Merritt (1992)], afterwards, a series of optimization follow-up works [Goldreich and Lindell (2006); Bellovin and Merritt (1993); Boyko, MacKenzie and Patel (2000); Shi (2018)].

Currently, a wide range of authenticated key exchange is password-based authentication key exchange (PAKE). In a nutshell, users in PAKE can generate a strong cryptographic common key using a shared "human-memorable" password without the helping of public-key infrastructure. The PAKE protocol can be divided to two types, symmetric PAKE and asymmetric PAKE, according to whether the passwords are same between the client and the server. If where the client and the server store different parameters, namely the client keeps a pwd but the server stores a one-way transformation v of the password, then this case is called asymmetric PAKE, otherwise, it is called symmetric PAKE. No matter which case, the two participants eventually reach agreement on a common key with high entropy using pwd and v (or pwd). Apparently, the main advantage of asymmetric PAKE is that, in case of server corruption, it prevents massive password recovering and it compels the attacker to recover passwords to perform an additional costly offline dictionary attack [Ford and Kaliski (2000)].

Two-factor authentication. Before proposing MFA schemes, a series of two-factor authentication (later denoted as TFA) schemes have been proposed that rely on knowledge (e.g., a password) and possession (e.g., a long cryptographic key) [Yang, Wong, Wang et al. (2006)]. TFA introduces a new factor such as one-time password and biometric etc. apart from the long-term password of user. Basically, TFA schemes are deployed by using a password of the user and a crypto-capable device. To our knowledge, many systems achieve authentication by adopting two-factor approach, such as one-time password PIN, including of SMS-based PIN and QR-code-based PIN, Google authenticator [WiK (2010a)], Due [Due (2019)], TOTP [WiK (2010c)], HOTP [WiK (2010b)]. More specifically, the client C authenticates to the server S by "proving possession" of an auxiliary physical device D (e.g., a smart-phone or a USB token) apart from know her password. The TFA scheme works as follows:

- Firstly, the device D displays a short one-time password PIN via either an SMS message or a QR code, where the PIN is either received from the server S (e.g., an SMS message or a QR code) or generated locally at D by using a secret key shared with S.
- Then, the user U manually types the PIN into his (or her) client terminal T along with his (or her) long-term password pwd.
- Finally, the server S determines to whether allow the user U to login or not by matching the received password pwd and PIN with the one stored in the database.

Multi-factor authentication. In order to establish a secure channel, Pointcheval et al.

[Pointcheval and Zimmer (2008)] first proposed the first MFAKE scheme using three factors (a password, a high entropy secret key and a biometric template) and obtained a common semantically secure secret key. In a nutshell, the client's authentication is the process where the clients access a remote server securely and the server executes the matching procedure based on a simple threshold on the Hamming distance between a reference template and the candidate template. For the next few years, various MFA schemes proposed to optimize the MFA scheme or launches attacks on the current MFA scheme then fixed the bug [Griffin and Phillip (2015) ; Portnoi and Shen (2016); Zhang, Xiao, Sun et al. (2017); Stebila, Udupi and Shantz (2008)].

Pythia PRF. The essence of the pythia PRF is a verifiable partially-obliviously pseudorandom function (PRF), e.g., SHA-256 and HMAC etc. To our knowledge, Everspaugh et al. [Everspaugh, Chatterjee, Scott et al. (2015)] introduced the pythia PRF service for the web server to keep the password security when the attacker launches brute-force cracking attacks on the password databases. Currently, most of web servers store the hashed password along with its salt instead of storing the plaintext of the password directly, but the trouble is that the attack can continue launch brute-force cracking attacks on the password databases. Thus, Everspaugh et al. [Everspaugh, Chatterjee, Scott et al. (2015)] introduced the pythia PRF to overcome these limitations. Afterwards, some follow-up optimized schemes depending on different requirements were proposed, such as Schneider et al. [Schneider, Fleischhacker, Schröder et al. (2016); Lai, Egger, Schröder et al. (2017); Jarecki, Krawczyk and Xu (2018c); Lai, Egger, Reinert et al. (2018)].

1.2 Paper organization

In Section 2 we review related notions. In Section 3, we give a high-level description for the system security model of our proposed protocol. In Section 4, we describe the proposed MFAKE via SPHF protocol, and the others building blocks which contains the short authentication string (SAS) message authentication protocol first proposed by Vaudenay [Vaudenay (2005)], password-to-random (PTR) protocol, and Pythia PRF services introduced by Everspaugh et al. [Everspaugh, Chatterjee, Scott et al. (2015)]. In Section 5, we conclude our contributions.

2 Preliminaries

In this section, we introduce required notations, definitions and lemmas.

Definition 1 (**Decisional Diffie-Hellman, (**DDH**)**). The decision-DDH assumption says that, in a group (p, $\mathbb{G}$, g), where we are given (g^a, g^b, g^c) for unknown random $a,b \leftarrow \mathbb{Z}_p$, it is hard to decide whether c=ab (mod q) (i.e., a real Diffie-Hellman tuple) or $c \xleftarrow{R} \mathbb{Z}_p$ (i.e., a random Diffie-Hellman tuple).

2.1 Fuzzy extractor

Fuzzy extractor was first proposed by Dodis et al. [Dodis, Reyzin and Smith (2004); Dodis, Ostrovsky, Reyzin et al. (2008)]. In this subsection, we give a brief view of the error tolerant fuzzy extractor.

Metric Space. We denote $\mathcal{M}$ is the metric set along with the distance hash dist: $\mathcal{M} \times \mathcal{M} \to \mathbb{R}^{+}=[0,+\infty)$. Notably, the hamming distance is used to another value.

Fuzzy Extractor. Inspired by the work of Dodis et al. [Dodis, Reyzin and Smith (2004); Dodis, Ostrovsky, Reyzin et al. (2008)], we can construct a fuzzy extractor given an error-correcting code (or ECC) and a universal hash family. In a nutshell, the fuzzy extractor is a two-tuple of polynomial-time algorithms (Gen, Rep), in more detail,

- $(\mathrm{P}, \mathrm{R}) \leftarrow \mathrm{Gen}(\mathrm{w})$ the algorithm takes as input the template $\mathsf{w} \in \mathcal{M}$, then outputs an extracted string $\mathsf{R} \in \{0,1\}^{\ell}$ which satisfy $\mathsf{SD}(\langle \mathsf{R},\mathsf{P}\rangle, \langle \mathsf{U}_{\ell},\mathsf{P}\rangle) \le \varepsilon$, where U_{ℓ} is uniform distribution on ℓ-bit binary strings and ε is negligible.
- $\mathsf{P} \leftarrow \mathsf{Rep}(\mathsf{w}',\mathsf{R})$, the Rep algorithm recovers the extracted string R from the corresponding public string P and any vector template w' that close to w.

Correctness. If there exist $\mathrm{dist}(\mathsf{w},\mathsf{w}') \le t$, then $\mathrm{Rep}(\mathrm{w}',\mathrm{R}) = \mathrm{R}$.

2.2 Biometric template

Below, we gave a brief of introduction of the biometric templates. As we know, biometric characteristics can be used as a unique identity.

However, the trouble is that there are no existing techniques to generate a biometric template such that the current template is same to the pre-extracted one. There exist some fuzzy matching techniques, e.g., Pointcheval et al. [Pointcheval and Zimmer (2008)], armed with these techniques, and some assumptions including of the encoding and Hamming distance, the matching decision can be formed as follows.

- The distance between two templates $\mathcal{T}_C$ and $\mathcal{T}_C^{'}$ of the same biometric is low with great (even overwhelmingly) probability. Rigorously, if for any client C, there exist the following equation

$$\Pr[\mathrm{dist}(\mathcal{T}_C, \mathcal{T}_C^{'}) \le t : \mathcal{T}_C^{'} \leftarrow \mathcal{D}_{B(C)}] \ge 1-\varepsilon_{fr},$$

where ε denotes the probability of "false rejection", further, for each client C, all his (or her) biometric templates form a probability biometric distribution $\mathcal{D}_{B(C)}$.

- The distance between $\mathcal{T}_C$ and $\mathcal{T}_C^{'}$ corresponding to two distinct clients C and C' is high with great (even overwhelmingly) probability. Formally, if for any pair of distinct clients $C \neq C'$, the probability of the distance between $\mathcal{T}_C$ and $\mathcal{T}_C^{'}$ is as follows:

$$\Pr[\mathrm{dist}(\mathcal{T}_C, \mathcal{T}_C^{'}) \ge \tau : \mathcal{T}_C \leftarrow \mathcal{D}_{B(C)}, \mathcal{T}_{C'} \leftarrow \mathcal{D}_{B(C')}] \ge 1-\varepsilon_{fa}$$

where $\tau > t$ is a threshold and ε_{fa} denotes the probability of "false acceptance".

2.3 ElGamal scheme

- $\mathsf{params} \leftarrow \mathsf{ElGamal}.\ \mathsf{Setup}(\lambda, G, q, g)$: Takes the security parameter λ, a cyclic group G with prime order q, i.e., $|q| = \lambda$, and the generator g of group G as

input, outputs the parameters $\mathsf{params} := (\lambda, G, q, g)$.

- $(sk, pk) \leftarrow \mathsf{ElGamal.KeyGen(params)}$: Takes the params as input, then samples a random $x \in_R \mathbb{Z}_q^*$. Outputs the secret key $sk := x$ and the public key $pk := (g, h = g^x)$.
- $\mathsf{ct} \leftarrow \mathsf{ElGamal.Enc}(pk, \mu)$: In order to encrypt the message μ, the algorithm first samples a random $r \in_R \mathbb{Z}_q$, then computes and outputs the ciphertext

$\mathsf{ct} := (c_1, c_2) = (g^r, h^r \cdot \mu)$

- $\mu \leftarrow \mathsf{ElGamal.Dec}(sk, \mathsf{ct})$: In order to decrypt the ciphertext, the algorithm computes and outputs $\mu := \frac{c_2}{c_1^x} = \frac{h^r \cdot \mu}{g^r x}$

It is well known that the ElGamal scheme is IND-CPA-secure under the decisional Diffie-Hellman assumption over G. Hence, we omit the further details.

3 System model

In this paper, we follow the security model of multi-factor password-based authentication key exchange (later denoted MFPAKE) proposed by Pointcheval et al. [Pointcheval and Zimmer (2008)], which is built upon the usual PAKE security model [Bellare and Rogaway (1994); Bellare, Pointcheval and Rogaway (2000)] in the real-or-random indistinguishable model [Abdalla, Fouque and Pointcheval (2005)]. In order to analyze the security of the proposed framework, we also assume that there exist a challenger who interacts with an adversary in the oracles, such as Test, Execute, Reveal, Send, and Corrupt defined in Pointcheval et al. [Pointcheval and Zimmer (2008); Jarecki, Krawczyk, Shirvanian et al. (2016)]. Further, in an MFA key exchange scheme, the advantage of the adversary is defined by $\mathsf{Adv}_{\mathrm{MFKE}}(\mathcal{A}) = |\Pr_{\mathcal{A}}^{\lambda}(\mathrm{Succ}) - 1/2|$. We omit the further details in this paper.

4 Our construction: multi factor authenticated key exchange

There is no doubt that passwords will continue to remain as a major authentication mechanism for humans in the foreseeable future. Thus, in this section, we give the detailed description for our multi-factor password-authenticated key exchange construction. In more detail, to finish the authentications for all three factors, password, one-time password (or secret key), and biometric, and to keep the privacy of biometric at the server side, i.e., the password and the biometric store at the password database of the registration center.

Before describing our framework, we first give a high-level description of our core idea. We observe the following issues:

- regarding for remote registration of client passwords, in the existing approaches, the client sends its password in plaintext to the server, while the server stores a transformation v derived from the received password (e.g., a hash value or a verifier (i.e., salt)) in a password database.
- the current Pythia PRF service approaches store the password at the password database by the web server along with the password and user identity and ignore the cases of

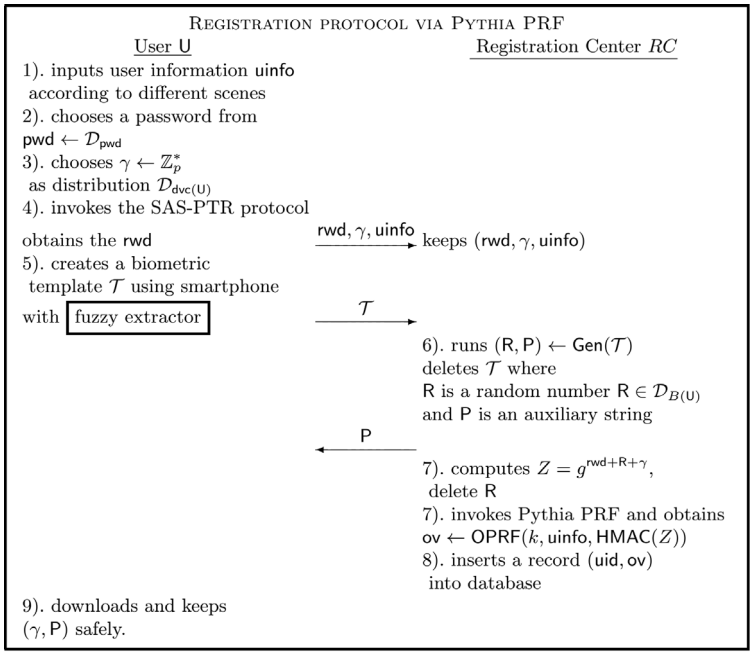
TFA and MFA.

- the current TFA and MFA schemes which use the biometric as one of factors only focus on how to prevent the leakage of the password and ignore the risks of the biometric leakage.

Thus, in this paper, to overcome the above-mentioned issues, we adopted the SAS-PTR protocol, Pythia PRF, and ElGamal encryption with an associate smooth projective hash function as the building blocks. To facilitate the understanding, we consider the following scenario. **1). Registration phase**. The user would register at the registration center (RC), but the user seeks to avoid inputting his low-entropy password pwd via the terminal directly, he leverages the SAS-PTR protocol (as shown in Fig. 2) to generate high entropy rwd. He then uses the rwd, his own biometric (e.g., fingerprint or iris) $\mathcal{T}$, and an associated one-time password γ as his factors to run the registration protocol (as shown in Fig. 1). To prevent the biological information leakage, the registration center invokes the Pythia PRF servers to harden the passwords rwd along with his corresponding biometric template. After that, the registration center inserts the hashed password and biometric into the database. **2). Login-authentication phase.** Finally, the user and the registration center execute the asymmetric PAKE protocol, as shown in Fig. 3, to generate high-entropy authentication key by running the variant ElGamal with an associated hash function.

4.1 Registration protocol via Pythia PRF

In order to present our protocol, we relax the requirement of the environment and only consider the registration takes place in a secure and reliable environment. The concrete registration protocol description is as follow the Fig. 1.



- The client can extract the (reproduced) secret $\mathsf{Rep}(\mathsf{P}, \mathcal{T}') \rightarrow \mathsf{R}$ using the biometrics $\mathcal{T}'$ and the auxiliary string P received from the *RC*.

Figure 1: Registration protocol via Pythia PRF

4.2 SAS-PTR protocol

In this subsection, we introduce the SAS-PTR protocol following the construction of Jarecki et al. [Jarecki, Krawczyk, Shirvanian et al. (2018a, 2018b)]. Here, the goal of SAS protocol is to assist the device to check the message integrity sent by the user. Meanwhile, the goal of PTR protocol is to obtain a high entropy password and avoid inputting the plaintext of password to the registration center. The concrete construction works as the following Fig. 2.

SAS AND PTR PROTOCOL

Parties. U: User; and D: Device.

Parameters. The protocol is parameterized by security parameter λ and SAS channel capacity t.

Contentment: Com_H is implemented via a hash function H with κ-bit output.

Protocol: User U sends the message $\mathsf{M_{UD}}$ to device D who verifies integrity of the received message via checksum received over the U-to-D channel.

1. U generates PKE key pair $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$, t-bit random value $\mathrm{str_U}$, and (2κ)-bit random value d, and random $r \leftarrow \mathbb{Z}_q$.
2. U generates $\mathsf{M_{UD}} = (pk, a)$ where $a = \mathsf{Hash}(\mathsf{pwd})^r$, then computes commitment $\mathsf{com} \leftarrow \mathsf{Com}_H(\mathsf{M_{UD}}, \mathrm{str_U}, d)$, and sends $(\mathsf{M_{UD}}, \mathsf{com})$ to D using text or QR code.
3. D samples random string $\mathrm{str_D}$ of length t and sends it to U.
4. upon receiving $\mathrm{str_D}$, U sends $(\mathrm{str_D}, d)$ to D and enters $\mathsf{checksum_U} \leftarrow \mathrm{str_U} \oplus \mathrm{str_D}$ into U-to-D SAS channel.
5. upon receiving $\mathsf{checksum_U}$ from the U-to-D SAS channel, D does the following steps:
   - D checks that received $\mathsf{checksum_D} \leftarrow \mathrm{str_U} \oplus \mathrm{str_D}$ is the same as the received $\mathsf{checksum_U}$. If so and also $\mathsf{com} = \mathsf{Com}_H(\mathsf{M_{UD}}, \mathrm{str_U}, d)$, D accepts $\mathsf{M_{UD}}$, else it rejects.
   - then D computes $b = a^k$, and sends $\mathsf{ct}_b := \mathsf{Enc}(pk, b)$ to U.
6. Lastly, the user U computes $b \leftarrow \mathsf{Dec}(sk, \mathsf{ct}_b)$, and obtains $\mathsf{rwd} \leftarrow \mathsf{H}_{\mathrm{oprf}}(\mathsf{pwd}, b^{1/r})$.

Figure 2: SAS and PTR protocol

In SAS-MA scheme, there exists two communication channels. One is open channel, and another is SAS channel. In a nutshell, the open channel is that allows the transmission of arbitrary length message and is controlled by an active man-in-the-middle attacker. Further, the SAS channel is that allows sending up to t bits that cannot be changed by the attacker. See more details from Jarecki et al. [Jarecki, Krawczyk, Shirvanian et al. (2018a)].

The PTR instantiation from Jarecki et al. [Jarecki, Krawczyk, Shirvanianet et al. (2016)] is based on the blind hashed Diffie-Hullman technique of Ford et al. [Ford and Kaliski (2000)]. The final goal of the PTR is that generates the $\mathsf{rwd} = F(k, \mathsf{pwd}) = H(\mathsf{pwd}, (H'(\mathsf{pwd}))^k)$ to the client.

4.3 Login-authentication protocol

- The client with uid uses a registered device and sends an authentication request to the server.
- The client inputs his (or her) password, then the client generates a new biometric template $\mathcal{T}'$ using the application of fuzzy extractor installed his (or her) smartphone. Afterwards, the clients reproduces the secret $\mathsf{R}' \leftarrow \mathsf{Rep}(\mathsf{P}, \mathcal{T}')$, where P was stored in

the client's smartphone. Finally, the client sends them $(\text{uid}, \mathsf{pwd}, \mathsf{R}')$ to the server.

- Upon receiving the tuple $(\text{uid}, \mathsf{pwd}, \mathsf{R}')$, the server recalls the stored γ and computes $Z = g^{\mathsf{rwd}+r+\mathsf{R}'}$, then the server leverages the Pythia PRF service to obtain the presudo random value ov'. Finally, the server will checks whether the received $\mathsf{ov}' \stackrel{?}{=} \mathsf{ov}$, where the ov stores in the database, if the verification is passed, then the user is allowed to login.

After finishing the stage of login-authentication, the client and the server begin to authenticate and generate the common key τ_C and τ_S via the key exchange.

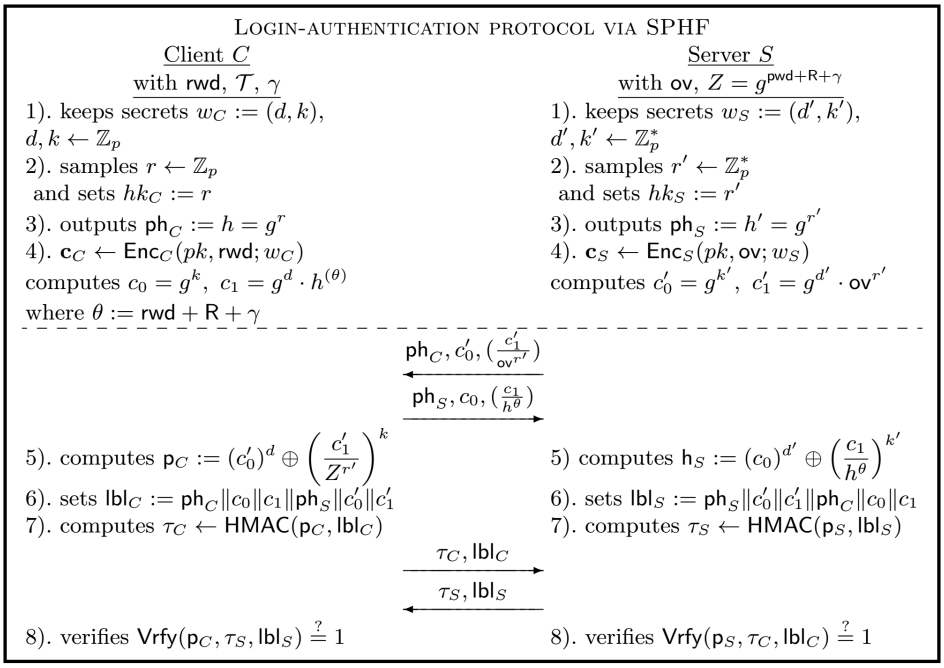


Figure 3: Login-authentication protocol via SPHF

4.4 Security analysis

The proposed MFAKE protocol is designed using the DDH-based building blocks including various ElGamal encryption, PRF, and PTR protocol, thus, the security of proposed MFAKE protocol can be proved easily if these building blocks are secure under the DDH and OMDG assumptions. Furthermore, the security of proposed combination of multi-factor password-authenticated key exchange protocol can be obtained easily depending on the universal composability methodology [Canetti (2001)]. We omit the further detailed proof in the current version that makes it easier to read.

4.5 Instantiate OPRF: Two-Hash DH-NIZK scheme

In this subsection, we describe how to instantiate the OPRF. In more detail, the main purposes of the receiver are to hash and blind her input and request to get the blind value

of the sender's secret-key application. Afterwards, the receiver attempts to verify the sender's response and then obtains the verifiable OPRF result via a second hash function. The associated verifiable oblivious PRF (vOPRF) protocol is simple:

- the client (or receiver) sends $a = H(x)^r$ to the sender and the sender after checking that $a \in \langle g \rangle$, it responds by $(y = g^k, b = a^k)$.
- the protocol terminates with the client returning the value $H_2(p, x, b^{1/r})$ after checking the tuple (g, y, a, b) is a valid DDH tuple, in this case, the π is regarded as $\pi := y = g^k$.
- further, the client achieves the latter test using a NIZK for equality of discrete logarithms. In more detail, the client shows that the tuple (g, y, a, b) whether satisfies the relation $\log_g^y = \log_a^b$. Oblivious, this is a standard protocol that we recall for completeness: the sender samples a random $t \in_R \mathbb{Z}_m$ to mask the secret (or witness) k and computes the challenge $c = H_3(g, y, a, b, g^t, a^t)$ as well as $z = r + c \cdot k \bmod m$. The proof is the pair $\zeta = (c, z)$ which we denote as $\mathsf{NIZK}_{EQ}^{H_3}[g, y, a, b]$ The receiver verifies $\zeta = (c, z)$ by testing $c = H_3(g, y, a, b, g^z y^{-c}, a^z b^{-c})$.

5 Conclusion

In this paper, we presented a new practical framework for MFA scheme which can be used to keep the privacy of biometric stored at server database. In our framework, we leverage the Pythia PRF service to harden stored biometric hashes (or ciphertexts) and the password hashes against offline brute-force attacks at the server side, and we use the smartphone as the accessibility tool to cooperate with the user and run the SAS protocol and the PTR protocol, which can enhance the security of the MFA scheme. In our framework, armed with the introduced the Pythia PRF service, the adversary cannot impersonate a client or a server, and cannot launch attack on the server database which store the user-identity, user's template, and user's random password. The security of the proposed MFA depends on the security of Pythia PRF, SAS protocol, and PTR protocol.

Acknowledgement: The authors would like to thank the anonymous reviewers for their helpful advice and comments. This work was supported by the National Natural Science Foundation of China (No. 61802214), the Natural Science Foundation of Shandong Province (Nos. ZR2019BF009, ZR2018LF007, ZR2017MF0, ZR2016YL011), the Shandong Provincial Key Research and Development Program of China (2018GGX1010052017, CXGC07012016, GGX109001), the Project of Shandong Province Higher Educational Science and Technology Program (No. J17KA049), and the Global Infrastructure Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2018K1A3A1A20026485).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Abdalla, M.; Fouque, P. A.; Pointcheval, D. (2005): Password-based authenticated key exchange in the three-party setting. *Public Key Cryptography-PKC*, vol. 3386, pp. 65-84.

Bellare, M.; Pointcheval, D.; Rogaway, P. (2000): Authenticated key exchange secure against dictionary attacks. *Advances in Cryptology-EUROCRYPT*, vol. 1807, pp. 139-155.

Bellare, M.; Rogaway, P. (1994): Entity authentication and key distribution. *Advances in Cryptology-CRYPTO' 93*, vol. 773, pp. 232-249.

Bellovin, S. M.; Merritt, M. (1992): Encrypted key exchange: password-based protocols secure against dictionary attacks. *Proceedings 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 72-84.

Bellovin, S. M.; Merritt, M. (1993): Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise. *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pp. 244-250.

Boyko, V.; MacKenzie, P.; Patel, S. (2000): Provably secure password-authenticated key exchange using Diffie-Hellman. *Advances in Cryptology-EUROCRYPT*, pp. 156-171.

Canetti, R. (2001): Universally composable security: a new paradigm for cryptographic protocols. *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pp. 136-145.

Due (2019): Secure two-factor authentication app. *Duo Mobile*. https://duo.com/product/trusted-users/two-factor-authentication/duo-mobile.

Dodis, Y.; Ostrovsky, R.; Reyzin, L.; Smith, A. (2008): Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, vol. 38, no. 1, pp. 97-139.

Dodis, Y.; Reyzin, L.; Smith, A. (2004): Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. *Advances in Cryptology-EUROCRYPT*, pp. 523-540.

Everspaugh, A.; Chatterjee, R.; Scott, S.; Juels, A.; Ristenpart, T. (2015): The pythia PRF service. *Proceedings of the 24th USENIX Security Symposium*, pp. 547-562.

Ford, W.; Kaliski, B. S. J. (2000): Server-assisted generation of a strong secret from a password. *IEEE International Workshops on Enabling Technologies*: *Infrastructure for Collaborative Enterprises*, pp. 176-180.

Goldreich, O.; Lindell, Y. (2006): Session-key generation using human passwords only. *Journal of Cryptology*, vol. 19, no. 3, pp. 241-340.

Griffin; Phillip, H. (2015): Biometric knowledge extraction for multi-factor authentication and key exchange. *Procedia Computer Science*, pp. 66-71.

Hao, F.; Clarke, D. (2012): Security analysis of a multi-factor authenticated key exchange protocol. *Cryptology ePrint Archive, Report 2012/039*.

http: //eprint.iacr.org/2012/039.

Jarecki, S.; Krawczyk, H.; Shirvanian, M.; Saxena, N. (2016): Device-enhanced password protocols with optimal online-offline protection. *ACM on Asia Conference on Computer & Communications Security*, pp. 177-188.

Jarecki, S.; Krawczyk, H.; Shirvanian, M.; Saxena, N. (2018a): Two-factor authentication with end-to-end password security. *Public-Key Cryptography-PKC*, pp. 431-461.

Jarecki, S.; Krawczyk, H.; Shirvanian, M.; Saxena, N. (2018b): Two-factor authentication with end-to-end password security and reduced user involvement. http://webee.technion.ac.il/~hugo/tfake.pdf.

Jarecki, S.; Krawczyk, H.; Xu, J. (2018c). OPAQUE: an asymmetric PAKE protocol secure against pre-computation attacks. *Advances in Cryptology-EUROCRYPT*, pp. 456-486.

Lai, R. W. F.; Egger, C.; Schröder, D.; Chow, S. S. M. (2017): Phoenix: rebirth of a cryptographic password-hardening service. *Proceedings of the 26th USENIX Security Symposium*, pp. 899-916.

Lai, R. W. F.; Egger, C.; Reinert, M.; Chow, S. S. M.; Maffei, M. et al. (2018): Simple password-hardened encryption services. *Proceedings of the 27th USENIX Security Symposium*, pp. 1405-1421.

Pointcheval, D.; Zimmer, S. (2008): Multi-factor authenticated key exchange. *International Conference on Applied Cryptography & Network Security*, vol. 5037, pp. 277-295.

Portnoi, M.; Shen, C. C. (2016): Location-enhanced authenticated key exchange. *International Conference on Computing, Networking and Communications*, pp. 1-5.

Shi, C. (2018): A novel ensemble learning algorithm based on D-S evidence theory for IoT security. *Computers, Materials & Continua*, vol. 57, no. 3, pp. 635-652.

Stebila, D.; Udupi, P.; Shantz, S. C. (2008): Multi-factor password-authenticated key exchange. *Cryptology ePrint Archive, Report 2008/214*, http://eprint.iacr.org/2008/214.

Schneider, J.; Fleischhacker, N.; Schröder, D.; Backes, M. (2016): Efficient cryptographic password hardening services from partially oblivious commitments. *ACM Sigsac Conference on Computer & Communications Security*, pp. 1192-1203.

Yang, G.; Wong, D. S.; Wang, H., Deng, X. (2006): Formal analysis and systematic construction of two-factor authentication scheme. *Cryptology ePrint Archive, Report 2006/270*, https://eprint.iacr.org/2006/270.

Vaudenay, S. (2005): Secure communications over insecure channels based on short authenticated strings. *Advances in Cryptology-CRYPTO*, pp. 309-326.

WiK (2010a): Google authenticator. *Wikipedia*.

https://en.wikipedia.org/wiki/Google_ Authenticator.

WiK (2010b): HMAC-based one-time password algorithm. *Wikipedia*.

https://en. wikipedia. org/wiki/ HMAC-based_One-time_Password_algorithm.

WiK (2010c): Time-based one-time password algorithm. *Wikipedia*,

https:// en.wikipedia.org/wiki/Time-based_One-time_Password_algorithm.

Zhang, R.; Xiao, Y.; Sun, S.; Ma, H. (2017): Efficient multi-factor authenticated key exchange scheme for mobile communications. *IEEE Transactions on Dependable and Secure Computing*, pp. 1.