cryptography

MDPI

Article

Efficient and Flexible Multi-Factor Authentication Protocol Based on Fuzzy Extractor of Administrator's Fingerprint and Smart Mobile Device

Alzahraa J. Mohammed and Ali A. Yassin *

Computer Science Department, Education College for Pure Sciences, University of Basrah, Basrah 6100, Iraq; alzahraa201593@gmail.com

* Correspondence: aliadel79yassin@gmail.com; Tel.: +964-7717542311

Received: 13 June 2019; Accepted: 5 September 2019; Published: 9 September 2019

Abstract: In an era of tremendous development in information technology and the Internet of Things (IoT), security plays a key role in safety devices connected with the Internet. Authentication is vital in the security field, and to achieve a strong authentication scheme, there are several systems using a Multi-Factor Authentication (MFA) scheme based on a smart card, token, and biometric. However, these schemes have suffered from the extra cost; lost, stolen or broken factor, and malicious attacks. In this paper, we design an MFA protocol to be the authenticated administrator of IoT's devices. The main components of our protocol are a smart mobile device and the fuzzy extractor of the administrator's fingerprint. The information of the authenticated user is stored in an anomalous manner in mobile devices and servers to resist well-known attacks, and, as a result, the attacker fails to authenticate the system when they obtain a mobile device or password. Our work overcomes the above-mentioned issues and does not require extra cost for a fingerprint device. By using the AVISPA tool to analysis protocol security, the results are good and safe against known attacks.

Keywords: IoT; MFA; administrator; authenticated server; smart mobile device; fuzzy extractor

1. Introduction

Computer networks and the Internet can be traced back to the 1960s and the late 1980s, respectively [1,2]. In the millennium era, mobile devices started to connect to the Internet via wireless/wireless networks [3], and, currently, the network connection, Internet, computer systems, websites, and mobile apps exist on multiple devices such as smart mobile phones, GPS devices, and others. Security risks represent one of the most critical challenges [4,5] that face computer systems and information technology, and access control is known as the core of security issues in the computer networks, which consists of authentication and authorization. It can allow office users to use the resources and services of the system in an authorized way and prevent illegal users from accessing the system's resources and services. Authentication is considered an essential component to protect the system, device or application from unlawful access—either in a direct or an indirect way [4–6]. To begin with, there was only one factor used to authenticate the users in the system; however, this approach can be easily compromised, particularly in the case of passwords [7,8]. In general, the user tends to use the same information for the accounts on different applications such as Facebook, Skype, and Gmail. An unauthorized user has the ability to compromise the account directly, and, moreover, an attacker can also try to apply well-known attacks such as the dictionary (online/offline) [9], social engineering, and gusting to access the sources and services of the system instead of the authorized user [10]. The password authentication schemes based on a single factor should be used for the minimum password complexity to protect the user's account from malicious attacks [11]. We discovered that the main problem is the memorizing of the password by the user.

In recent years, Two-Factor Authentication (2FA) [12–14] was presented for use of a second factor—such as an SMS token, token device, smart card, and biometric with the user's account information—for resisting malicious attacks and solving the memorizing issue in password complexity [15–17].

While the security approaches grow in the 2FA field, attackers' methods also increase. Although 2FA schemes are strong, they still suffer from malicious attacks including lost/stolen smart card attacks, taking a fake fingerprint from the original fingerprint, and insider attacks. There is therefore a need for a more secure and strong scheme based on the Multi-Factor Authentication (MFA) to check the validity of users. MFA represents the power source to protect the system against an unauthorized user and reduce the risks of malicious attacks. Principally, MFA consists of various factors such as biometrics (behavioral and biological characteristics), the smart mobile device, token device, and smart card. This type of authentication scheme increases the security degree and allows for the application of identification, verification, and authentication for ensuring user authority. Figure 1 demonstrates the authentication schemes. Currently, MFA is considered a vital part in many fields in the information technology world and is involved in processes such as validating the identity of the administrator of the system, IoT devices, and smart mobile devices (see Figure 2).

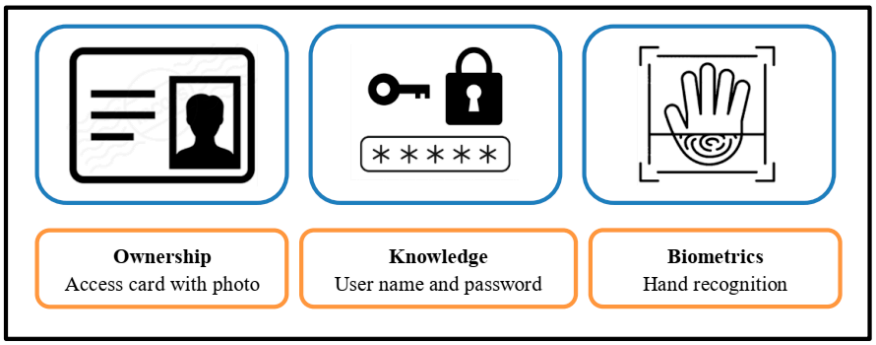


Figure 1. Conceptual authentication examples.

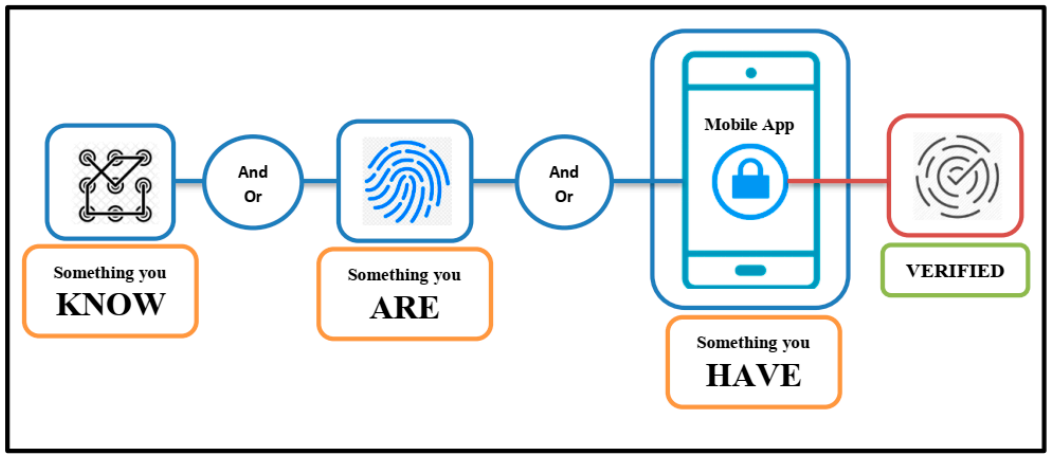


Figure 2. Multi-factor authentication.

The major advantage for using MFA schemes is to increase security level—"*assume an adversary can get user's password by applying malicious attacks but he faces difficulties to retrieve information of another factor*" [17]. Furthermore, some MFA schemes use a token device that does not need coverage for a mobile network or the Internet.

MFA does have several drawbacks, such as forgotten/stolen mobile phone, smart card or token device, allowing illegal access to the authorized user's accounts and extra cost for detection resources for the hardware's maintenance centers. For that reason, our work aims to propose a strong and

lightweight protocol for user authentication, based on a fuzzy extractor of fingerprints and smart mobile phone for administrators of IoT devices and server–client systems, among others. The important contributions of our proposed protocol are as follows:

- We design a secure and robust protocol for authenticating the legal administrator that can play the main role for management of IoT devices in future work or server–client systems. This work has positive features such as mutual authentication, once-secure session key for each login phase, the anonymity of password and biometric, unlinkability, and security against a stolen smart mobile device and online change of password.
- We propose an MFA protocol based on a fuzzy extractor of the user's fingerprint, smart mobile phone and an encrypted-credential file that keeps our main factors in safe mode against malicious attacks.
- We present a security analysis of our work, and we have noted that our protocol resists the replay attacks, man-in-the-middle attacks, insider attacks, offline dictionary attacks, eavesdropping and traffic attacks.
- Consequently, we prove the proposed protocol on standard and strong security proof tool AVISPA.
- We present the comparative analysis of our proposed protocol with other related work based on security analyses and resisting well-known attacks.

The contents of this paper are as follows. Section 2 presents a related work. Section 3 describes the symbols and cryptography concepts used in our protocol. The details of our proposed protocol are outlined in Section 4. Section 5 discusses the security analysis of our protocol. Section 6 reveals the experimental results using the AVISPA tool. Section 7 refers the comparison with other related works. Section 8 concludes our research.

2. Related Work

The traditional methods of authentication are to use a PIN code, a password, etc. [18], and the password here represents knowledge management of the main factor. The researchers then added the second factor to increase the security in the form of a physical token that contributes to strengthening user authentication schemes, e.g., a smart card [19,20], and other devices such as a smartphone.

A one-time password method is used to generate one password for each user login request. This method is applied in many systems [21]; however, its main problem is repetition and it is uncontrollable. In recent years, authors have proposed MFA schemes, and they began by introducing biometrics (physical and behavioral) into authentication schemes—for instance, voice biometrics, face recognition, methods of eye recognition (iris recognition and retinal recognition), hand geometry, fingerprint, electrocardiographic (ECG) recognition, electroencephalographic (EEG) recognition, DNA recognition, etc. There was a problem with regards to biological agents, which was difficult to modify, and this caused security issues in the system.

Chen et al. [22] proposed a scheme by using two main tools: (1) secured mobile phones; and (2) fingerprint of the mobile's user. Their work focused on capturing the front-end and back-end fingerprint recognition system in smart mobile phones. Derawi et al. [23] presented a scheme relying on cameras of smart mobile phones to obtain fingerprint images instead of a scanner device. Moreover, Sin et al. [24] suggested that a system consisted of two phases: the first one is used to build users' templates while the second checks the validity of the user based on their template. This scheme has been applied in the commercial market since 2009. Ravi and Sivanath [25] also presented a scheme that applies a camera to obtain the user's finger image and then computes the background and feature extraction of the fingerprint to authenticate the user.

Dhillon et al. [26] proposed a remote-user authentication protocol for the IoT with three factors—passwords, smart cards and biometrics. Park and Park [13] proposed a biometric authentication system and used a fuzzy extractor, smartcard and elliptic curve cryptosystem (ECC). Their work resists attacks and uses BAN logic to provide a mutual authentication.

However, these factors can be stolen, lost or broken, thus researchers turned to the use of smart phones, especially after they developed the ability to extract the feature of biometrics. Some authors have suggested using mobile devices in the validation process. Buschek et al. [27] provided a tapping process on the smart phone screen; since the typing pattern is unique to each person, this approach can be used as a validating agent [28,29]. Belk et al. [30] proposed a paper about the difference between conventional passwords and realistic ones in terms of efficiency and effectiveness, where the latter takes longer. Michelin et al. [31] suggested a facial and iris recognition system by using the smartphone's camera. Jeong et al. [32] developed a scheme combining the smartphone, password, and biometric parameters of the user—creating an MFA system. Nevertheless, the disadvantage of this system is that it deals with limited security threats, and the accuracy of recognition technology based on biometric was not good. Sun et al. [33] presented a scheme based on user biometrics with smartphone information using a fuzzy extractor. This system does not store raw information for biometric measurements, thus, depending on the offline method to change the biometrics or password, it does resist some attacks.

In this paper, we propose the MFA protocol based on a fuzzy extractor of user's fingerprint and a smart mobile phone. Our work can resist malicious attacks and has good features such as mutual authentication, once-secure session key for each login phase, the anonymity of password and biometric, unlinkability, securing against the stolen smart mobile device, and online change of password.

3. Symbols Used and Cryptography Concepts

3.1. Symbols Used

Some basic symbols are used in the search (Table 1).

Table 1. Basic Symbols.

Symbol	Description
AS	Authenticated Server.
Adm_i	The administrator of a system.
SMD_i	Smart mobile device.
m	The number of administrators.
ID_{A_i}	Identity of administrators.
ID_{AS}	Identity of an authenticated server.
Pw_{A_i}	The password of an administrator.
FP_{A_i}	Fingerprint of administrators.
r_0, r_1, r_2	Random integer number.
Z_M	Set of integer numbers within range M.
R	Secret information of Fuzzy Extractor.
P	Public Reproduction parameter of Fuzzy Extractor.
K_{A_i}	Secret parameter of Administrator.
$Gen()$	Generator function of Fuzzy extractor.
$Rep()$	Reproduction function of Fuzzy extractor.
$h()$	Hash function.
$Enc()$	Encryption function.
$Dec()$	Decryption function.
IF_{AS}	Index file of an administrator.
CF_{SMD_i}	Credential file of Smart mobile device.
N_{A_i}	Contain sensitive information.
s	The private key of an administrator.
M, v, w	Large prime integer numbers.
$\|$	Concatenation operation.
$\|^{-1}$	The inverse of Concatenation operation.
$*$	Multiplication operation.

Table 1. *Cont.*

Symbol	Description
G	Group of *DDH*.
g	The generator of the group.
q	A large prime number.
T_{A_i}	Time of an administrator.
T_{AS}	Time of an authenticated server.
X_{A_i}	A secret number of an administrator.
X_{AS}	A secret number of an authenticated server.
ΔT	Timestamp.
SK	Session key

3.2. Cryptography Concepts

- *The decisional Diffie–Hellman (DDH)* key exchange protocol [34]: A and B agree on a finite cyclic group G and choose a generator g from them. They then choose randomly a,$b \in [1, |G|]$ and exchange g^a and g^b. The secret key is g^{ab}. To break the protocol, a passive eavesdropper, i.e., attacker (Eve), must compute the DDH function, defined as $DH_g(g^a, g^b) = g^{ab}$.
- *Fuzzy Extractor*: The fuzzy extractor is a cryptography method for securely authenticating using biometric. Suppose a finite set L is a metric space with a distance function *dis* along with an error limit, E, calculated using error correction codes (Hamming distance, set difference metric, edit distance metric, etc.) [35,36]. The fuzzy extractor contains two operations i.e., Generator (*Gen*) and Reproduction (*Rep*), with the following features:
    - The $Gen()$ operation takes a biometric $B_i \in L$ of user U_i as input and outputs a secret string $\sigma_i \in \{0.1\}^l$, and a public string $p \in \{0.1\}^*$, i.e., $\text{Gen}(B_i) = (\sigma_i, p)$.
    - The $Rep()$ operation takes a noisy biometric $B_i' \in L$ of user U_i and the public string p as input and reproduces the secret string $\sigma_i \in \{0.1\}^l$ as an output, i.e., $Rep(B_i', \tau_i) = \sigma_i$ if and only if dis $(B_i, B_i') \leq T$.
- *Hash function (MD5)* [37]: The cryptographic hash function is MD5 (message digest 5). MD5 generates a 128-bit message digest of the input, which is expressed as a 32-digit hexadecimal number. MD5 hash outputs are unique even if the size of the input is different.
- *Encryption/Decryption by AES algorithm* [38]: The input and output for the AES algorithm contain sequences of 128 bits (digits with values of 0 or 1). These sequences are referred to as blocks and the number of bits they contain as their length. The cipher key for the AES algorithm is a sequence of 128, 192 or 256 bits. Other input, output and cipher key lengths are not permitted by this standard.

4. Our Proposed Protocol

In this section, we propose a strong multi-factor authentication protocol based on the password and the user's fingerprint in the Internet server system and depended on an application in a smart mobile device. Our protocol includes three main elements—Authenticated Server (*AS*) such as cloud service provider, Administrator ($Adm_1 \ \ldots \ Adm_m$), and Smart Mobile Device (SMD_i)—and is divided into four phases: registration, login, authentication, and change password. The registration phase is implemented only once, while the login and authentication phases are performed whenever Adm_i needs to log into the system, and the change password phase works at the moment when Adm_i wishes to change the password.

4.1. Registration Phase

During this phase, our work depends on two steps: the first is related to Adm_i that manages the system while the second connects with *AS* (see Figure 3).

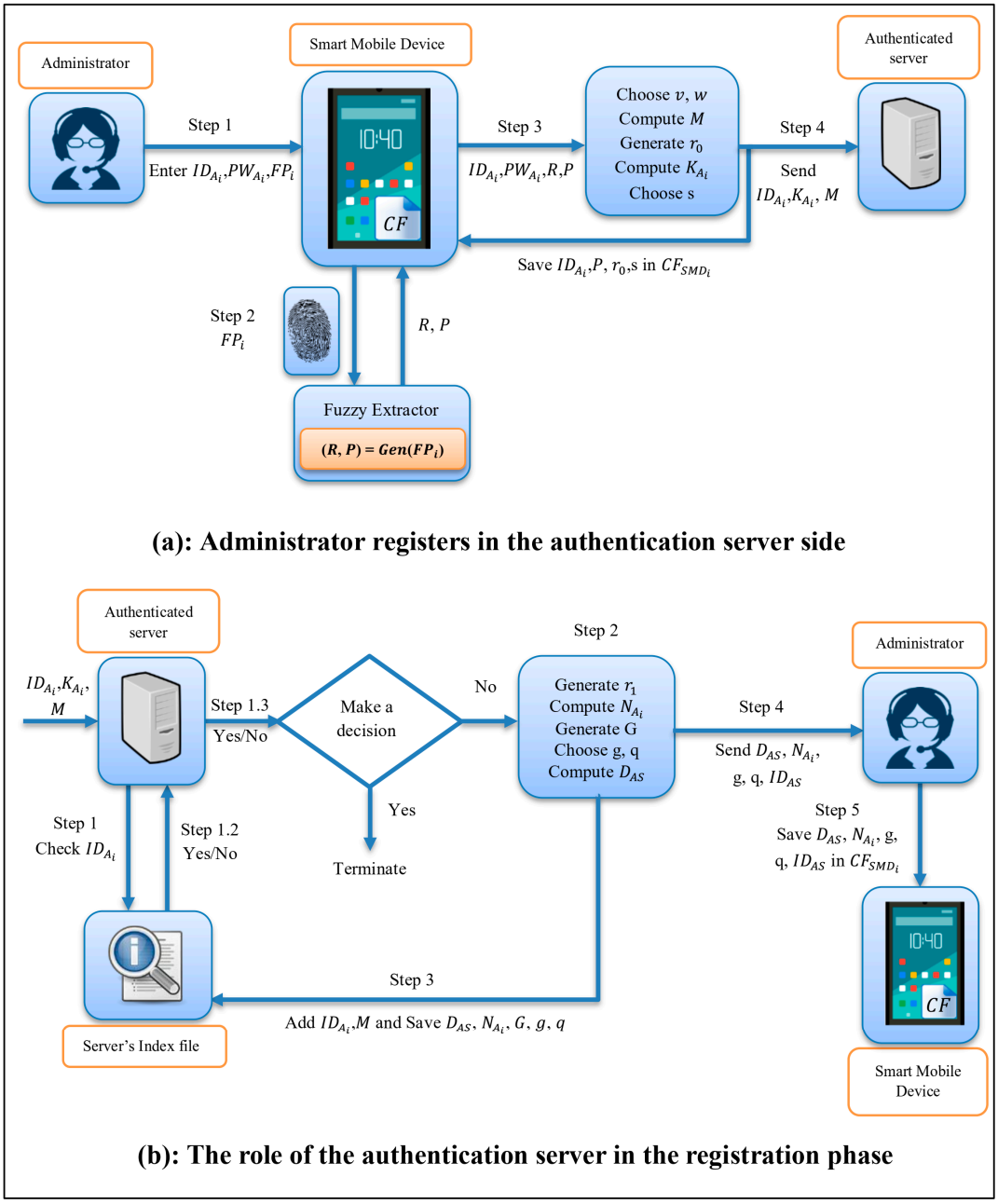


Figure 3. Registration phase.

StepR1: Administrator Side:

- Adm_i should be registered into AS, they depend on their SMD_i to complete this process via mobile apps, and this step can be described in the following points:
- Adm_i enters their Username (ID_{A_i}), Password (PW_{A_i}), and Fingerprint (FP_{A_i}) by using a mobile app.
- After that, SMD_i applies the generator function of the fuzzy extractor to extract (R,P) from FP_{A_i}, (R,P) = $Gen(FP_{A_i})$.
- Adm_i chooses two large prime integer numbers v, w ϵ Z and computes $M = v * w$.
- Adm_i generates a random integer number r_0 ϵ Z_M, computes a secret parameter $K_{A_i} = h(h(R,r_0), ID_{A_i}, PW_{A_i})$ and chooses a private key (s) (note: s is used with things and users).
- After that, Adm_i creates a Credential file (CF_{SMD_i}) to save the main parameters (ID_{A_i}, r_0, P, s, v, w) into SMD_i secretly based on $CF'_{SMD_i} = CF_{SMD_i} \oplus R \oplus PW_{A_i}$.

- Adm_i uses their SMD_i to send a message $M_1 = (ID_{A_i}, K_{A_i}, M)$ to AS (see Figure 3a).

StepR2: Authenticated Server Side:

Upon receiving a message from Adm_i, AS implements the following steps:

- AS checks the identity of Adm_i in Index File (IF_{AS}) and compares ID'_{A_i} (receiving) $\stackrel{?}{=} ID_{A_i}$ (existing) in IF_{AS}; if the result is equal, Adm_i has been registered in the system. Otherwise, they create a new record to save ID_{A_i} of Adm_i in IF_{AS}.
    - AS generates a random integer number $r_1 \in Z_M$, computes $N_{A_i} = h(r_1, ID_{A_i}, K_{A_i})$, and adds (r_1, N_{A_i}) to Adm_i's record in IF_{AS}, where N_{A_i} contains sensitive information used in the next phases by Adm_i.
    - Based on the decisional Diffie–Hellman (DDH) assumption, AS generates a group (G) and chooses a generator g from the group and a large prime number (q), computes $D_{AS} = g^{K_{A_i}} \bmod q$ and saves (M, g, q, D_{AS}) in IF_{AS}.
- AS sends $M_2 = (N_{A_i}, D_{AS}, g, q, ID_{AS})$, to Adm_i (see Figure 3b).

StepR3: Administrator Side:

When the Adm_i receives M_2 from AS, they decrypt their $CF_{SMD_i} = CF'_{SMD_i} \oplus R \oplus PW_{A_i}$ and add M_2 to it. After that, they encrypt $CF'_{SMD_i} = CF_{SMD_i} \oplus R \oplus PW_{A_i}$.

4.2. Login Phase

StepL1: Administrator Side:

When Adm_i wishes to log into the system, they should perform the following steps (see Figure 4):

- They enter Identity (ID_{A_i}), Password (PW_{A_i}), and Fingerprint (FP_{A_i}) by using a mobile device SMD_i to allow them to use the important parameters inside CF_{SMD_i}.
- SMD_i applies the reproduction function of the fuzzy extractor to calculate $R' = Rep\ (P, FP_{A_i})$.
- SMD_i decrypts $CF'_{SMD_i} = CF_{SMD_i} \oplus R' \oplus PW_{A_i}$ to allow Adm_i to use the important parameters inside CF_{SMD_i}.
- SMD_i verifies the authority of Adm_i based on extracting important parameters from the above steps and CF_{SMD_i} as follows:
    - SMD_i computes $K'_{A_i} = h(h(R', r_0), ID_{A_i}, PW_{A_i})$, $D'_{AS} = g^{K'_{A_i}} \bmod q$.
    - After that, SMD_i compares D'_{AS} (computing) $\stackrel{?}{=} D_{AS}$ (existing) in the credential file; if the result is false, they terminate the login phase. Otherwise, SMD_i performs the following steps:
- SMD_i generates a random integer number $r_2 \in Z_M$, computes $h_{A_i} = h(T_{A_i} \| r_2 \| ID_{A_i})$.
- SMD_i computes ASCII code for h_{A_i}, $y_{A_i} = \sum_{i=1}^{length(h_{A_i})} ASCII\left(h_{A_i}(i)\right)$, $V_{A_i} = \left(g^{y_{A_i}}\right)^{X_{A_i}} \bmod q$, and $U_{A_i} = (V_{A_i} \| D_{AS})$.
- They send $M_3 = (ID_{A_i}, (ID_{AS} \| r_2), U_{A_i}, T_{A_i})$ to AS.

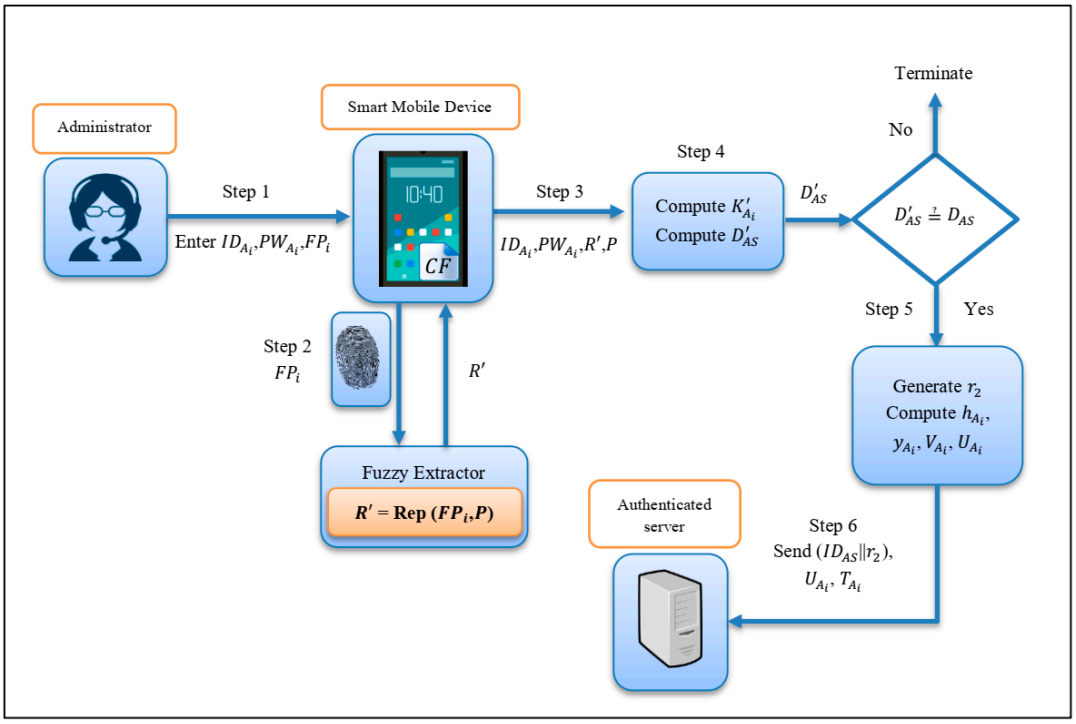


Figure 4. Login phase of the administrator.

StepA1: Authenticated Server Side:

Upon receiving a message M_3 from Adm_i, AS implements the following steps:

- AS checks time stamp $T_{AS} - T_{A_i} \leq \Delta T$; if the equivalent condition is not fulfilled, AS aborts the authentication phase. Otherwise, AS compares $ID'_{AS} \stackrel{?}{=} ID_{AS}$; if they match, AS calculates the following steps, otherwise AS terminates the authentication phase.
  - AS computes $h_{AS} = h\ (T_{A_i}||r_2||ID_{A_i})$ and applies ASCII code for h_{AS}, $y_{AS} = \sum_{i=1}^{length(h_{AS})} ASCII(h_{AS}(i))$.
  - AS computes $V_{AS} = (g^{y_{AS}})^{X_{AS}} \bmod q$, the session key $SK_{AS} = {V_{A_i}}^{X_{AS}} \bmod q$, $P_{AS} = (V_{AS}||N_{A_i})$, and encrypts $(ID_{A_i}||P_{AS})$ by using SK:

$$I_{AS} = Enc_{SK}\left(ID_{A_i}||P_{AS}\right)$$

- Finally, AS sends $M_4 = (I_{AS}, P_{AS}, T_{AS})$ to Adm_i.

StepA2: Administrator Side:

After Adm_i receives M_4 from AS, Adm_i computes the following steps:

- They check $T_{A_i} - T_{AS} \leq \Delta T$; if not equal, they abort the authentication phase. Otherwise, they verify N_{A_i} (from CF_{SMD_i}) $\stackrel{?}{=} N'_{A_i}$ (from P_{AS}); if not equal, they terminate the authentication phase. Otherwise, they implement these steps (see Figure 5):
  - Adm_i computes the session key $SK = {V_{AS}}^{X_{A_i}} \bmod q$, from decrypts I_{AS} by using SK to retrieve $(ID'_{A_i}, P'_{AS}) = Dec_{SK}(I_{AS})$.

- After that, Adm_i checks if P_{AS} (from decrypted I_{AS}) $\stackrel{?}{=} P'_{AS}$ (from M_4) and ID_{A_i} (from decrypted I_{AS}) $\stackrel{?}{=} ID'_{A_i}$ (existing); if not equal, they terminate the process, otherwise they compute $J_{A_i} = (ID_{AS} \| D_{AS})$ and encrypt $(ID_{A_i} \| J_{A_i})$ by using SK:

$$M_5 = Enc_{SK}\left(ID_{A_i} \| J_{A_i}\right).$$

- Finally, Adm_i sends M_5 to AS.

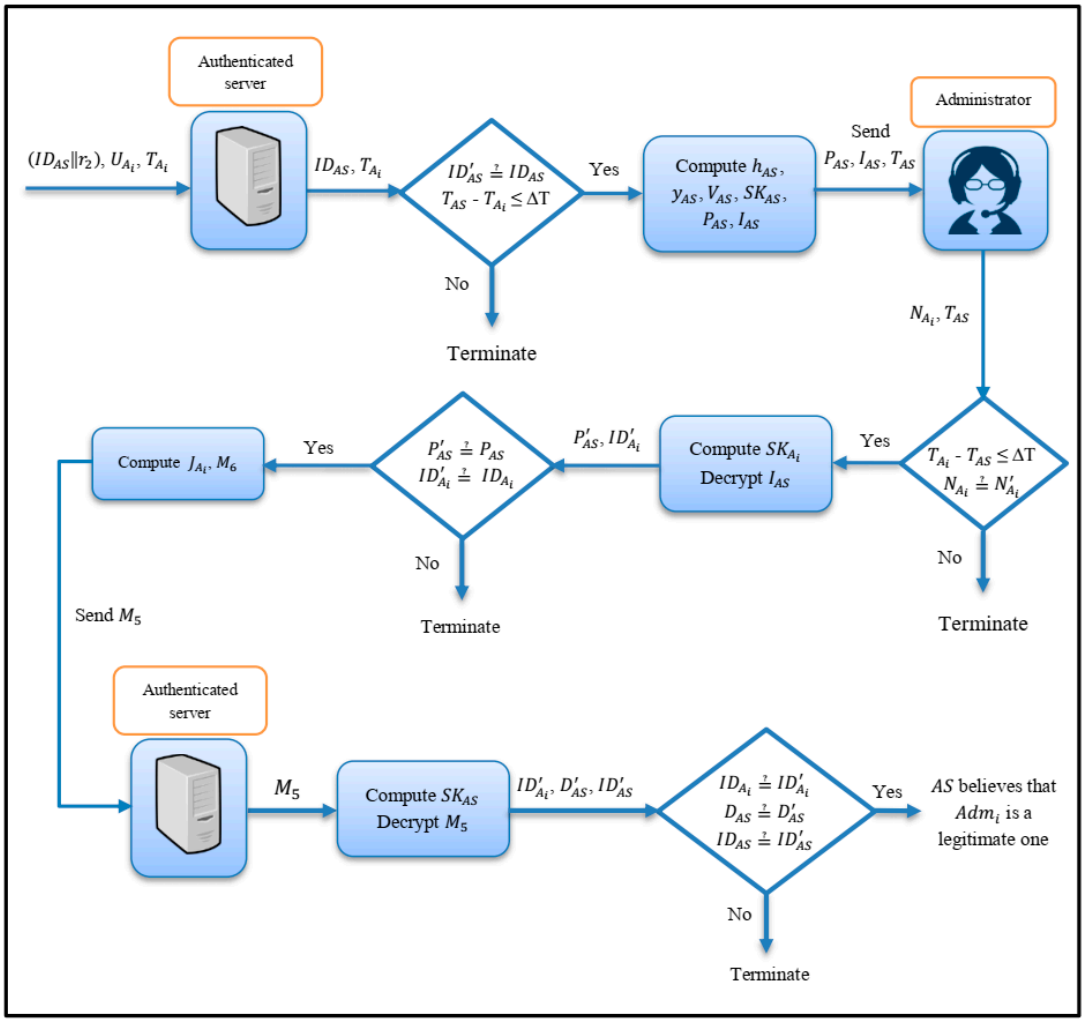


Figure 5. Authentication phase between authenticated server and administrator.

StepA3: Authenticated Server Side:

Finally, AS receives the message from Adm_i and computes the following steps:
AS computes the session key $SK = {V_{A_i}}^{X_{AS}} \bmod q$ to decrypt M_5 to retrieve $(ID'_{A_i}, D'_{AS}, ID'_{AS}) = Dec_{SK}(M_5)$ and check $ID_{A_i} \stackrel{?}{=} ID'_{A_i}$, $D_{AS} \stackrel{?}{=} D'_{AS}$, $ID_{AS} \stackrel{?}{=} ID'_{AS}$; if the results are true, AS believes that Adm_i is legitimate and can manage their devices and access to the services and resource of AS. Otherwise, AS terminates this phase.

4.3. Change Password Phase

StepCP1: Administrator Side:

When Adm_i wants to change their fingerprint (FP_{A_i}) and Password (PW_{A_i}) to a new fingerprint (FP'_{A_i}) and a new Password (PW'_{A_i}), the following steps are performed (see Figure 6):

- Adm_i enters the specified ID_{A_i}, PW_{A_i}, FP_{A_i} using SMD_i to log into the system.
- SMD_i computes $R' = Rep(P, FP_{A_i})$, and decrypts $CF'_{SMD_i} = CF_{SMD_i} \oplus R' \oplus PW_{A_i}$ to allow Adm_i to use important parameters inside CF_{SMD_i}.
- SMD_i checks ID_{A_i} of Adm_i with the value stored in CF_{SMD_i} when ID_{A_i} is correcting, they compute $K'_{A_i} = h(h(R', r_0), ID_{A_i}, PW_{A_i})$, $D'_{AS} = g^{K'_{A_i}} \bmod q$ and compare $D_{AS} \stackrel{?}{=} D'_{AS}$ (existing) in CF_{SMD_i}; if not equal, they terminate the password change request, otherwise they go to the next steps.
- Adm_i enters the new fingerprint (FP'_{A_i}) and new Password (PW'_{A_i}) into SMD_i, which computes the new values $(R'', P') = Gen(FP'_{A_i})$, $K''_{A_i} = h(h(R'', r_0), ID_{A_i}, PW'_{A_i})$, $D''_{AS} = g^{K''_{A_i}} \bmod q$.
- Eventually, SMD_i updates the values (D''_{AS},P') in CF_{SMD_i}, encrypts $CF'_{SMD_i} = CF_{SMD_i} \oplus R'' \oplus PW_{A_i}$, and sends $M_6 = (K''_{A_i}, (ID_{AS} \| D''_{AS}), T_A, ID_{A_i})$.

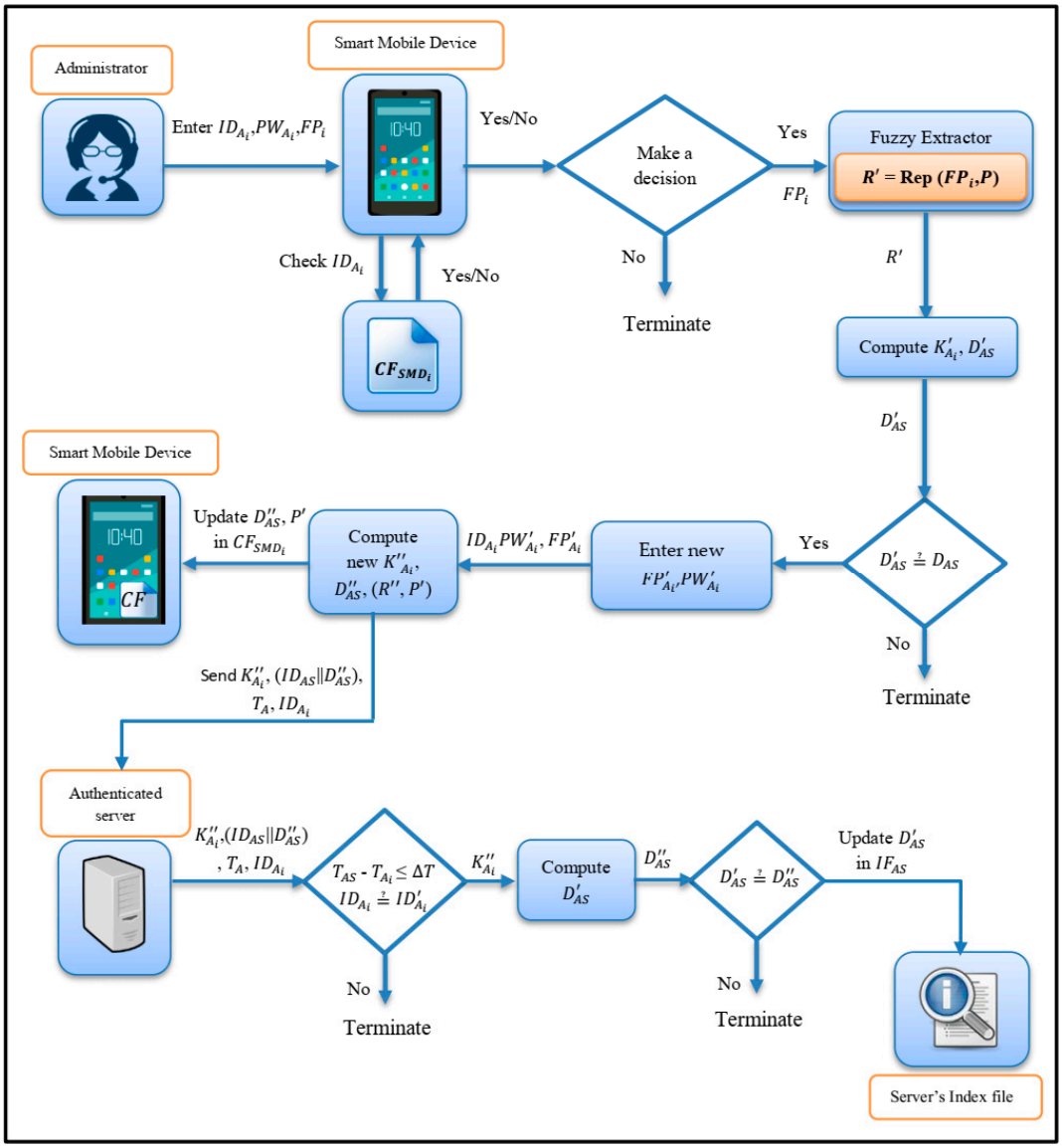


Figure 6. Change password phase.

StepCP2: Authenticated Server Side:

When AS receives M_6 from Adm_i, AS follows these steps:

- AS checks $T_{AS} - T_{A_i} \leq \Delta T$, ID_{A_i} (from IF_{AS}) $\stackrel{?}{=} ID'_{A_i}$ (from M_6); if not equal, they terminate the request.
- AS computes new $D'_{AS} = g^{K''_{A_i}} \bmod q$, and checks D'_{AS} (computing) $\stackrel{?}{=} D''_{AS}$ (from M_6); if the equation is not equal, the process ends. Otherwise, they update the value of D_{AS} in IF_{AS} with D'_{AS}.

5. Security Analysis

In this part, we prove that our proposed protocol is safe and secure against well-known malicious attacks such as eavesdropping and traffic attacks. Additionally, the proposed protocol has provided robust features such as mutual authentication, password anonymity, and secure session key and we support the comparative analysis of related authentication protocols. As a result, our work has been verified based on AVISPA Tool, denoting that our protocol is secure and safe (see Tables 2–7).

Proposition 1. *Our protocol provides mutual authentication.*

Proof. Mutual authentication means both Administrator (Adm_i) and Authenticated Server (AS) validate each other. Our protocol therefore focuses on three components (Adm_i, SMD_i, and AS) to achieve this feature. □

In the login phase side, Adm_i wants to log into the system based on their mobile device SMD_i applying the following steps:

1. SMD_i checks the validity of Adm_i's parameters (ID_{A_i}, PW_{A_i}, and FP_{A_i}) by applying the fuzzy extractor function to compute $D'_{AS} = g^{K'_{A_i}} \bmod q$.
2. Then, SMD_i compares $D'_{AS} \stackrel{?}{=} D_{AS}$ (existing in the credential file); if the result is false, Adm_i is not authorized and terminates the login phase. Otherwise, SMD_i computes $V_{A_i} = (g^{y_{A_i}})^{X_{A_i}} \bmod q$, $U_{A_i} = (V_{A_i} \| D_{AS})$.
3. They send $M_3 = ((ID_{AS} \| r_2), U_{A_i}, T_{A_i})$ to AS.

Upon receiving M_3, the authentication phase is started with the first step, abd AS ensures the authority of Adm_i as follows:

1. AS checks time stamp $T_{AS} - T_{A_i} \leq \Delta T$; if so, they compare $ID'_{AS} \stackrel{?}{=} ID_{AS}$, and, if not a match, they ensure the Adm_i is not authorized. Otherwise, they compute the following steps: AS computes $V_{AS} = (g^{y_{AS}})^{X_{AS}} \bmod q$, the session key $SK = V_{A_i}{}^{X_{AS}} \bmod q$, and $P_{AS} = (V_{AS} \| N_{A_i})$, and encrypts $(ID_{A_i} \| P_{AS})$ by using SK:

$$I_{AS} = Enc_{SK}\left(ID_{A_i} \| P_{AS}\right) \tag{1}$$

2. AS sends $M_4 = (I_{AS}, P_{AS}, T_{AS})$ to Adm_i.

After Adm_i receives M_4 from AS, Adm_i computes the following steps:

1. They check $T_{A_i} - T_{AS} \leq \Delta T$ and N_{A_i} (from CF_{SMD_i}) $\stackrel{?}{=} N'_{A_i}$ (from P_{AS}); if the results are not equal, they know the AS is unauthorized and terminate the authentication phase. Otherwise, they implement these steps: Adm_i decrypts I_{AS} by using SK to retrieve $(ID'_{A_i}, P'_{AS}) = Dec_{SK}(I_{AS})$. After that, Adm_i checks P'_{AS} (from decrypted I_{AS}) $\stackrel{?}{=} P_{AS}$ (from M_4), ID'_{A_i} (from decrypted I_{AS}) $\stackrel{?}{=} ID_{A_i}$ (existing); if the results are equal, they terminate the process, otherwise they compute $J_{A_i} = (ID_{AS} \| D_{AS})$ and encrypt $(ID_{A_i} \| J_{A_i})$ by using SK:

$$M_5 = Enc_{SK}\left(ID_{A_i} \| J_{A_i}\right). \tag{2}$$

2. Adm_i sends M_5 to AS as a second factor.

Finally, AS retrieves the $(ID'_{A_i}, D'_{AS}, ID'_{AS}) = Dec_{SK}(M_5)$ and checks $ID_{A_i} \stackrel{?}{=} ID'_{A_i}$, $D_{AS} \stackrel{?}{=} D'_{AS}$ and $ID_{AS} \stackrel{?}{=} ID'_{AS}$; if the results are true, AS believes that Adm_i is legitimate and can manage their devices and access to the services and resource of AS. Otherwise, AS terminates this phase. Therefore, our proposed protocol has mutual authentication feature in a secure manner. Table 7 explains the safety of mutual authentication in a practical sense, based on AVISPA.

Proposition 2. *Our protocol provides a once-secure session key.*

Proof. In our protocol, the session key is used between Adm_i and AS in the login and authentication phases, which plays a vital role in the encrypting or decrypting of exchanging messages between Adm_i and AS. The main equation to create a once-secure session key is as follows:

$$SK = {V_{A_i}}^{X_{AS}} \bmod q \tag{3}$$

where V_{A_i} is computed by Adm_i based on $y_{A_i} = \sum_{i=1}^{length(h_{A_i})} ASCII\left(h_{A_i}(i)\right)$, $V_{A_i} = (g^{y_{A_i}})^{X_{A_i}} \bmod q$, and $U_{A_i} = (V_{A_i} || D_{AS})$. Then, Adm_i sends $((ID_{AS} || r_2), U_{A_i}, T_{A_i})$ to AS. □

In the authenticated server side, AS ensures the validity information of Adm_i and then retrieves $V_{A_i} = (U_{A_i} \|^{-1} D_{AS})$ and applies the above equation depending on secret parameter X_{AS}. After that, they use SK to encrypt $Enc_{SK}(ID_{A_i} || P_{AS})$ and send (P_{AS},I_{AS}, T_{AS}) to Adm_i. Finally, Adm_i calculates ($SK = {V_{AS}}^{X_{A_i}} \bmod q$), where V_{AS} is computed by AS based on $y_{AS} = \sum_{i=1}^{length(h_{AS})} ASCII(h_{AS}\ (i))$, $V_{AS} = (g^{y_{AS}})^{X_{AS}} \bmod q$, and $P_{AS} = (V_{AS} || N_{A_i})$. From P_{AS} and decrypted I_{AS}, Adm_i verifies the information received from AS. The r_2 mainly contributes to generating one SK for each login phase. Assuming an adversary tries to obtain SK, they fail to access secure parameters (r_2, X_{A_i}, X_{As}, h_{A_i}, h_{AS}, y_{A_i}, y_{AS}, g, and q). As a result, our proposed protocol provides a once-secure session key and the experimental results in Table 7 refer to the secrecy of SK.

Proposition 3. *Our protocol provides anonymity of password and biometrics.*

Proof. When the administrator starts to register in the system, they use their main parameters based on their Identity (ID_{A_i}), Password (PW_{A_i}), and Fingerprint (FP_{A_i}). They compute the following points:

- SMD_i applies the generation function of the fuzzy extractor on FP_{A_i}, extracts the main values $(R, P) = Gen(FP_{A_i})$, generates a random integer number $r_0 \in Z_M$, and computes $K_{A_i} = h(h(R,r_0), ID_{A_i}, PW_{A_i})$.
- Administrator's parameters (R, K_{A_i}) are not saved in AS and CF_{SMD_i} while each pair of (P, r_0) is saved in CF_{SMD_i} and other parameters (ID_{A_i}, D_{AS}, N_{A_i}, and M) are saved as anomalous elements in AS to check the validity of Adm_i in the login and authentication phases (where $N_{A_i} = h(r_1, ID_{A_i}, K_{A_i})$, $D_{AS} = g^{K_{A_i}} \bmod q$). Assuming an attacker has the ability to access the main parameters (P, r_0, D_{AS}, N_{A_i}, and M), the attacker cannot know the details of Adm_i or AS as these parameters have been saved in an anomalous way and they fail to use it again to login instead of Adm_i. □

Proposition 4. *Our protocol provides unlinkability.*

Proof. This feature confirms that an administrator may try multiple logins to the server to use resources/services without others being able to link these logins together [39]. In our proposed protocol, each time Adm_i wants to log into the system they submit $M_3 = ((ID_{AS} || r_2), U_{A_i}, T_{A_i})$ to AS. Thus, the primitive components of M_3 are generated once for each login phase by using the following points:

- SMD_i generates a random integer number $r_2 \in Z_M$ and computes $h_{A_i} = h(T_{A_i} \| r_2 \| ID_{A_i})$.
- SMD_i computes ASCII code for h_{A_i}, $y_{A_i} = \sum_{i=1}^{length(h_{A_i})} ASCII\left(h_{A_i}(i)\right)$, $V_{A_i} = \left(g^{y_{A_i}}\right)^{X_{A_i}} \bmod q$, and $U_{A_i} = (V_{A_i} \| D_{AS})$. □

As a result, the primitive parameters of M_3 generate once and AS cannot link many logins with the same Adm_i.

Proposition 5. *Our protocol is secure against stolen smart mobile devices.*

Proof. When a smart mobile device (SMD_i) is stolen, an attacker cannot use the SMD_i unless they know the device password or biometric factor. However, assuming the attacker succeeds in obtaining the device password and accessing the application, they will not be able to use it as the application needs Identity (ID_{A_i}), Password (PW_{A_i}), and a live Fingerprint (FP_{A_i}) for the account owner in the application (the authorized user). We refer that the credential file was saved in an encrypted way $CF'_{SMD_i} = CF_{SMD_i} \oplus R' \oplus PW_{A_i}$. Furthermore, it is difficult to decrypt CF'_{SMD_i} depending on the password and fingerprint of Adm_i and an attacker fails to have any advantages from stolen smart mobile device. □

Proposition 6. *Our protocol is resistant to replay attacks.*

Proof. The attacker takes the information and sends it later without modification. Supposing the attacker intercepts messages $M_3 = ((ID_{AS} \| r_2), U_{A_i}, T_{A_i})$ and $M_4 = (I_{AS}, P_{AS}, T_{AS})$ and tries to use it to log into the system. This login has one result that an attacker fails to use these parameters as the time is terminated and $((T_{AS} - T_{A_i} \leq \Delta T), (T_{A_i} - T_{AS} \leq \Delta T))$ are not achieved and the other values (r_2, U_{A_i}, I_{AS}, and P_{AS}) have been generated only once. Therefore, our protocol is safe against replay attacks. □

Proposition 7. *Our protocol is sturdy against Man-In-The-Middle (MITM) attacks.*

Proof. MITM is intercepting a conversation between the parties to the communication; the conversation appears normal for both parties, however, all the information exchanged passes through the attacker, and they can eavesdrop or modify and re-send. We assume that the attacker has obtained $M_3 = ((ID_{AS} \| r_2), U_{A_i}, T_{A_i})$ and modified it as $M_3{}^* = (\left(ID_{AS} \| r_2\right)^*, U_{A_i}{}^*, T_{A_i}{}^*)$; the modified message does not work, as the AS verifies the $ID_{AS}{}^*$ that was sent by the Adm_i, and finds that ($ID_{AS} \neq ID_{AS}{}^*$). In addition, AS cannot get the value of (r_2) from $\left(ID_{AS} \| r_2\right)^*$. Additionally, the message M_3 is generated once for each login phase. Thus, our protocol does not allow MITM attacks. □

Proposition 8. *Our protocol is resistant to an insider attack.*

Proof. This type of attack means that an authorized person has the ability to access the system and apply some negative changes. We assume that an authorized person (Adm_1) wants to obtain another authorized person's device (Adm_2) to access their account (Adm_2) or (Adm_1) using (Adm_2)'s account in an unauthorized manner. According to our protocol, (Adm_1) cannot do this attack, as the application needs a live fingerprint (FP_{A_i}) of the (Adm_2), ensuring that the owner of the original device cannot be impersonated. In addition, the file is protected against stolen/used device as it is encrypted $CF'_{SMD_i} = CF_{SMD_i} \oplus R \oplus PW_{A_i}$ with a value R that requires a live fingerprint to extract (R,P) = $Gen(FP_{A_i})$. As a result, our protocol is resistant to an insider attack. □

Proposition 9. *Our protocol is resistant to eavesdropping and traffic attacks.*

Proof. This is the process of intercepting and examining messages to extract information from them. All messages exchanged between the Adm_i and AS are the parameters used only once (r_2, U_{A_i}, I_{AS},

P_{AS}, and J_{A_i}), thus, if the eavesdropping and traffic attacks intercept these parameters, the attackers fail to enter the system. □

- Adm_i sends $M_3 = ((ID_{AS}||r_2), U_{A_i}, T_{A_i})$ to AS.
- AS sends $M_4 = (I_{AS}, P_{AS}, T_{AS})$ to Adm_i.
- Adm_i sends $M_5 = Enc_{SK}(ID_{A_i}||J_{A_i})$ to AS.

Note that the messages M_3, M_4, and M_5 are generated once for each admin's login request. Accordingly, our protocol is resistant to eavesdropping and traffic attacks.

Proposition 10. *Our protocol can resist the offline dictionary attack.*

Proof. The dictionary here is a list of words that attackers believe to be used by Adm_i in the formulation of their password. In our protocol, collecting login's messages does not help the attacker to predicate $Adm_i's$ password. We assume that an attacker catches values during each of Adm_i's login requests (r_2, h_{A_i}, y_{A_i}, $V_{A_i} = (g^{y_{A_i}})^{X_{A_i}} \bmod q$, $U_{A_i} = (V_{A_i}||D_{AS})$), which are generated once by SMD_i, and these values cannot be used again. As a result, our protocol is resistant to an offline dictionary attack. □

Proposition 11. *Our protocol provides an online change of password.*

Proof. Our protocol allows the administrator to change their old fingerprint and old password to a new one. This is done only when there is a connection between Adm_i and AS, where Adm_i logins to the system with the old fingerprint and old password, SMD_i computes $R' = Rep(P, FP_{A_i})$, decrypts CF'_{SMD_i}, and checks ID_{A_i} of Adm_i. After that, Adm_i computes K'_{A_i}, D'_{AS}, and compares $D_{AS} \stackrel{?}{=} D'_{AS}$ (existing) in CF_{SMD_i}, if not equal, they terminate the password change request; otherwise, they enter the new fingerprint (FP'_{A_i}), new password (PW'_{A_i}) and SMD_i compute the new values $(R'', P') = Gen(FP'_{A_i})$, $K''_{A_i} = h(h(R'',r_0), ID_{A_i}, PW'_{A_i})$, $D''_{AS} = g^{K''_{A_i}} \bmod q$. Eventually, SMD_i updates the values (D''_{AS}, P') in CF_{SMD_i}, encrypts $CF'_{SMD_i} = CF_{SMD_i} \oplus R'' \oplus PW'_{A_i}$, then send $M_6 = (K''_{A_i}, (ID_{AS}||D''_{AS}), T_A, ID_{A_i})$ to AS to update it after confirming the identity of Adm_i. Therefore, our protocol provides security in the case of an insider attacker or loss of the device. □

When AS receives M_6 from Adm_i, AS checks $T_{AS} - T_{A_i} \leq \Delta T$, ID_{A_i} (from IF_{AS}) $\stackrel{?}{=} ID'_{A_i}$ (from M_6), if not equal, they terminate the request.

AS computes new $D'_{AS} = g^{K''_{A_i}} \bmod q$, and checks D'_{AS} (computing) $\stackrel{?}{=} D''_{AS}$ (from M_6), if the equation is not equal, they finish the process. Otherwise, they update the value of D_{AS} in IF_{AS} with D'_{AS}.

6. Experimental Results

To implement and simulate the presented protocol on AVISPA, we focused on the main tool called Security Protocol Animator (SPAN) Version 1.6 on a computer system containing Windows 10 Enterprise operating system (64 bit), supported by Ubuntu 10.10 light on Virtual machine, Intel (R) Core (TM) i7-7500U CPU @ 2.70 GHz 2.90 GHz processor, and 8 GB RAM. We executed our proposal protocol considering a minimal number of components included in Server–Client/IoT (i.e., administrator, authentication server, and device) based on Dolev–Yao model with a restricted number of sessions, detected goal, On-the-Fly Model-Checker (OFMC) and Constraint-Logic based Attack Searcher (CL-AtSe) backend [40].

The AVISPA Tool

The AVISPA tool is one of the new techniques used to analyze and study the security of protocols used. It is a generally accepted and strong software tool for automatically authenticating (depending

on push-button technique) the security characteristics of the protocols used in Server–Client/Internet of Things (see Figure 7).

The protocol was implemented in HLPSL (the High-Level Protocol Specification Language) language, and, after the protocol as written, a HLPSL2IF translator converted this code into an Intermediate Format (IF) [41,42]. The steps of back-ends are as follows:

1. On-the-fly model-checker (OFMC)
2. Constraint-logic based attack searcher (CL-AtSe)
3. SAT-based model-checker (SATMC)
4. Tree automata based on automatic approximation for the analysis of security protocols (TA4SP)

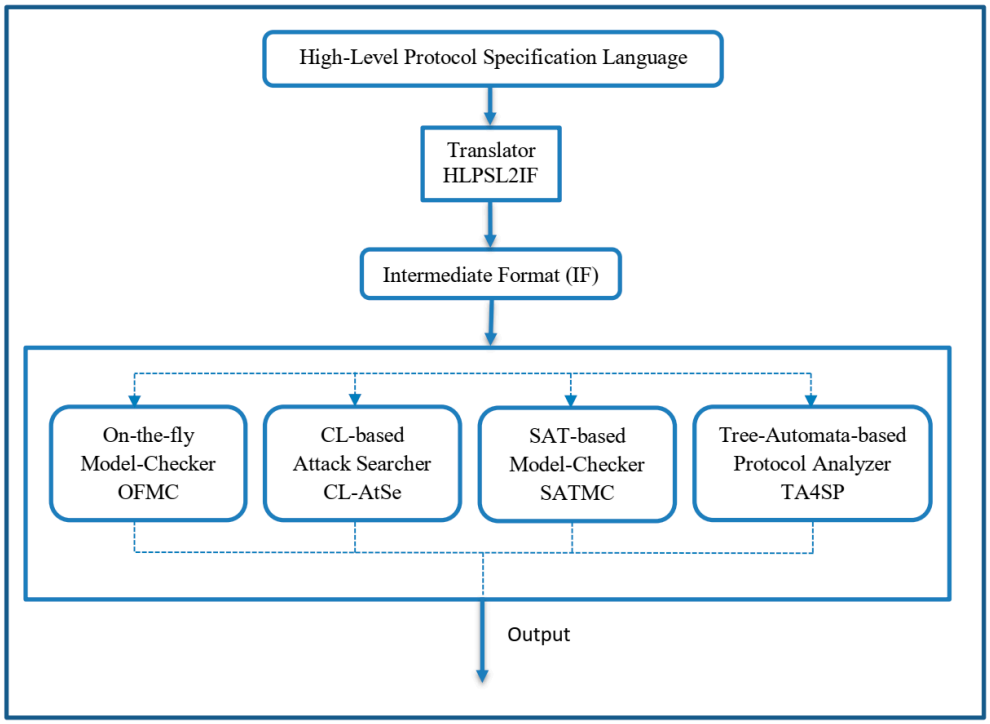


Figure 7. AVISPA Architecture.

Based on HLPSL specification, our proposed protocol consists of:

1. **Basic Role** explains the activity of the objects (e.g., Authenticated Server *AS* and Administrator of a system Adm_i) contained in the protocol (see Tables 2 and 3).
2. **Transitions** are defined as steps. The first basic role starts with a statement containing the beginning. This status only changes after receiving a message.
3. **Composed Roles** contain one or more basic roles to implement together and denote the sessions involved in the protocol (see Table 4).
4. **Environment** contains all sessions; the attacker may perform some roles as an authorized user (see Table 5).
5. **Security Goal** defines the security objective of the protocol. Some of the goals used in this section include:

    - Secret (SK', sk, {Adm,AS}): It specifies that the information (SK') is secretly shared with Adm and AS.
    - Witness (Adm, AS, admin_authserver_ra, Ra2'): It represents the weak authenticity of Adm by AS and Adm is the witness for the data Ra2'. The identity of this goal is represented as admin_authserver_ra in the goal section.

- Request (Adm, AS, admin_authserver_m, M6′): It represents the strong authenticity of Adm by AS on M6′ with an identity admin_authserver_m.

Table 2. Specification of Adm_i's role in HLPSL.

```
role admin ()
Adm,AS: agent,
S: symmetric_key,
H,Gen,Rep,Mul,DDH,ASCII,Enc,Dec: hash_func,
SND,RCV: channel(dy))
played_by Adm def=
local
State: nat,
Das,Das1,FP,FP1,G,G1,Ha,Has,Ias,IDa,IDas,Ja,Ka,Ka1,M,M6,Na,Pas,PFP,PWa,Q,Ra1,Ra2,Ras1,RFP,RFP1,Ta,Ta1,
Tas,Ua,V,Va,Vas,W,Xa,Xas,Ya,Yas: text,
SK: message
const admin_authserver_sk,authserver_admin_sk,authserver_admin_ra,admin_authserver_ra,adminv,authsv:
protocol_id
init
State:= 0
transition
0.State = 0 /\ RCV(start) =|>
State′:= 2 /\ V′:= new()
/\ W′:= new()
/\ M′:= Mul(V′.W′)
/\ Ra1′:= new()
/\ RFP′:= Gen(FP)
/\ Ka′:= H(H(RFP′,Ra1′),IDa,PWa)
/\ S′:= new()
/\ secret(S′,adminv,Adm)
/\ SND(IDa,Ka′,M′)

2.State = 2 /\ RCV(Na′,Das′,G1′,Q′,IDas′) =|>
State′: = 4 /\ RFP1′: = Rep(FP1.PFP)
/\ Ka1′: = H(H(RFP1′, Ra1),IDa,PWa)
/\ Das1′: = DDH (exp (G1′, Ka1′). Q′)
/\ Ra2′: = new ()
/\ Xa′: = new ()
/\ secret(Xa′, adminv,Adm)
/\ Ta′: = new ()
/\ Ha′: = h (Ta′.IDa.Ra2′)
/\ Ya′: = ASCII(Ha′)
/\ Va′: = exp(DDH(exp(G1′,Ya′).Q′),Xa′)
/\ secret (Va′,adminv,Adm)
/\ Ua′: = (Va′.Das′)
/\ witness (Adm,AS,admin_authserver_ra,Ra2′)
/\ SND ((IDas′.Ra2′),Ua′,Ta′)

4.State = 4 /\ RCV(Ias′,Pas′,Tas′) =|>
State′:= 6 /\ Ta1′:= new()
/\ Vas′:= Mul(Pas′.Na)
/\ SK′:= DDH(exp(Vas′,Xa),Q)
/\ Ja′:= (IDas.Das)
/\ M6′:= {IDa.Ja′}_SK′
/\ secret(SK′,sk,{Adm,AS})
/\ request(Adm,AS,admin_authserver_m,M6′)
/\ SND(M6)
/\ request(Adm,AS,admin_authserver_sk,SK′)
end role
```

Table 3. Specification of *AS*'s in HLPSL.

```
role authserver ()
Adm,AS: agent,
S: symmetric_key,
H,Gen,Rep,Mul,DDH,ASCII,Enc,Dec: hash_func,
SND,RCV: channel(dy))
played_by AS def=
local
State: nat,
Das,Das1,FP,FP1,G,G1,Ha,Has,Ias,IDa,IDas,Ja,Ka,Ka1,M,M6,Na,Pas,PFP,PWa,Q,Ra1,Ra2,Ras1,RFP,RFP1,Ta,Ta1,
Tas,Ua,V,Va,Vas,W,Xa,Xas,Ya,Yas: text,
SK: message
const admin_authserver_sk, authserver_admin_sk, admin_authserver_m, authserver_admin_m,
authserver_admin_ra, admin_authserver_ra, adminv, authsv: protocol_id
init
State:= 1
transition
1.State = 1 /\ RCV(IDa,Ka',M') =|>
State':= 3
/\ Ras1': = new()
/\ Na': = H(Ras1'.IDa.Ka')
/\ G': = new()
/\ G1': = new()
/\ Q': = new()
/\ Das': = DDH(exp(G1',Ka').Q')
/\ secret(Ras1',authsv,AS)
/\ SND(Na',Das',G1',Q',IDas)

3.State = 3 /\ RCV((IDas'.Ra2'),Ua',Ta') =|>
State': = 5 /\ Tas': = new()
/\ request (Adm,AS,authserver_admin_ra,Ra2')
/\ Has': = h(Ta'.IDa.Ra2')
/\ Yas': = ASCII(Has')
/\ Xas': = new()
/\ Vas': = exp(DDH(exp(G1,Yas').Q),Xas')
/\ secret(Xas',authsv,AS)
/\ Va': = Mul(Ua'.Das)
/\ SK': = DDH(exp(Va',Xas').Q)
/\ Pas': = (Vas'.Na)
/\ Ias': = {IDa.Pas'}_SK'
/\ secret(SK',sk,{AS,Adm})
/\ SND (Ias',Pas',Tas')
/\ witness(AS,Adm,authserver_admin_sk,SK')

5.State =5 /\ RCV(M6') =|>
State': = 7 /\ SK': = DDH(exp(Va,Xas).Q)
/\ request(AS,Adm,authserver_admin_m,M6')
/\ IDas': = {M6'}_SK
end role
```

Table 4. Specification of proposed protocol's session in HLPSL.

```
role session ()
Adm,AS:agent,
S: symmetric_key,
H,Gen, Rep, Mul, DDH, ASCII, Enc, Dec: hash_func)
def=
local SAdm, RAdm, SAS, RAS: channel(dy)
composition
admin(Adm,AS,S,H,Gen,Rep,Mul,DDH,ASCII,Enc,Dec,SAdm,RAdm)
/\ authserver (Adm,AS,S,H,Gen,Rep,Mul,DDH,ASCII,Enc,Dec,SAS,RAS)
end role
```

Table 5. Specification of proposed protocol's environment in HLPSL.

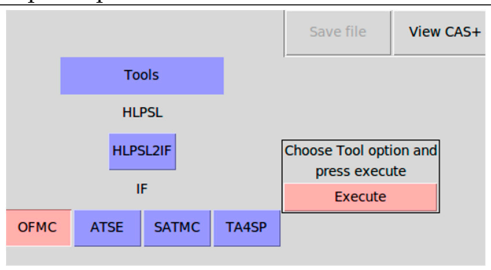
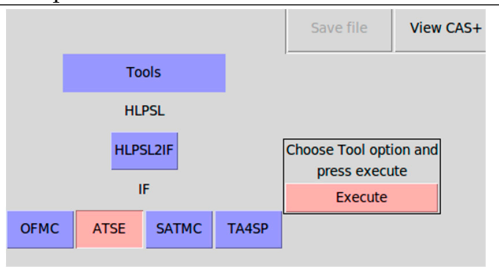
```
role environment ()
def=
const
admin_authserver_m,authserver_admin_m,
adminv,authsv,sk,
authserver_admin_ra,admin_authserver_ra,
admin_authserver_sk,authserver_admin_sk: protocol_id,
adm, as: agent,
sadminas,sias,sadmini: symmetric_key,
h,gen, rep,mul,ddh,ascii,enc,dec: hash_func

intruder_knowledge = {adm,as,h,gen,rep,mul,ddh,ascii,enc,dec,sias,sadmini}
composition
session(adm,as,sadminas,h,gen, rep,mul,ddh,ascii,enc,dec)
/\ session(i,as,sias,h,gen, rep,mul,ddh,ascii,enc,dec)
/\ session(adm,i,sadmini,h,gen, rep,mul,ddh,ascii,enc,dec)
end role
```

Table 6. Specification of proposed protocol's goal in HLPSL.

```
goal
secrecy_of sk,adminv,authsv
authentication_on
admin_authserver_m,authserver_admin_m,authserver_admin_ra,admin_authserver_ra,admin_authserver_sk,
authserver_admin_sk end goal
environment()
```

Table 7. Security verification result obtained using the AVISPA tool.

Using OFMC BACKEND	**Using CL-ATSE BACKEND**
SUMMARY	SUMMARY
SAFE	SAFE
DETAILS	DETAILS
BOUNDED_NUMBER_OF_SESSIONS	BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL	TYPED_MODEL
/home/span/span/testsuite/results/testify	PROTOCOL
GOAL	home/span/span/testsuite/results/test.if
as specified	GOAL
BACKEND	As Specified
OFMC	BACKEND
COMMENTS	CL-AtSe
STATISTICS	STATISTICS
parseTime: 0.00 s	Analysed: 0 states
searchTime: 0.16 s	Reachable: 0 states
visitedNodes: 12 nodes	Translation: 0.04 s
depth: 4 plies	Computation: 0.00 s
	

Description of the Output of AVISPA Tool

The output created by the AVISPA tool contains the following sections (see Table 7):

- **Summary:** It specifies the security reliability of the protocol regarding safe, unsafe or inconclusive.
- **Details:** The output determines the environment and context under which the protocol is claimed to be safe, unsafe or inconclusive.
- **Protocol:** The name of the protocol required for documentation is written here.
- **Goal:** This section represents the specified security goal of the protocol.
- **Backend:** This section denotes one of the four back-ends.

7. Comparison with Other Related Works

7.1. Security Features

We compared the security features of our proposed protocol with some protocols from previous studies, as shown in Table 8.

Table 8. Security features comparison.

	Security Feature	[12]	[13]	[40]	[43]	[44]	Our Protocol
1	Provides mutual authentication.	Yes	Yes	Yes	Yes	Yes	Yes
2	Provides a once secure session key.	Yes	Yes	Yes	Yes	Yes	Yes
3	Provides anonymity of password and biometric.	Yes	Yes	Yes	Yes	Yes *	Yes
4	Provides unlinkability.	No	No	No	No	No	Yes
5	Secure against Stolen Smart Mobile Device.	No	No	No	No	No	Yes
6	Resistant to replay attacks.	Yes	No	Yes	Yes	Yes	Yes
7	Resistant to Man-in-the-middle attacks.	No	No	Yes	No	No	Yes
8	Resistant to eavesdropping and traffic attacks.	No	No	Yes	No	No	Yes
9	Resistant to an offline dictionary attack.	Yes	Yes	Yes	Yes	Yes	Yes
10	Resistant to an insider attack.	Yes	Yes	Yes	Yes	Yes	Yes
11	Provides an online change password.	No	No	Yes	No	Yes	Yes

Yes *: means the feature that is not complete.

7.2. Performance Comparisons

We compared the calculation costs of the phases (registration, login and authentication) in our proposed work with previous work (one-way cryptographic hash function (T_h), fuzzy extractor used in biometric verification (T_f), and symmetric key encryption/decryption (T_E)), as shown in Table 9.

Table 9. Computational cost comparison.

	Registration Phase		Authentication and Login Phase		Total	
	Adm_i	AS	Adm_i	AS	Adm_i	AS
Our Protocol	$2\,T_h + T_f + 3\,T_E$	T_h	$3\,T_h + T_f + 2\,T_E$	$T_h + 2\,T_E$	$5\,T_h + 2\,T_f + 3\,T_E$	$2\,T_h + 2\,T_E$
[12]	$T_h + T_f$	$3\,T_h$	$10\,T_h + T_f + T_E$	$10\,T_h + 2\,T_E$	$11\,T_h + 2\,T_f + T_E$	$13\,T_h + 2\,T_E$
[13]	$T_h + T_f$	$5\,T_h$	$9\,T_h + T_f$	$11\,T_h$	$10\,T_h + 2\,T_f$	$16\,T_h$
[40]	T_h	$4\,T_h$	$3\,T_h$	$3\,T_h$	$4\,T_h$	$7\,T_h$
[43]	$4\,T_h + T_f$	$2\,T_h$	$6\,T_h + T_f + T_E$	$3\,T_h + 2\,T_E$	$10\,T_h + 2\,T_f + T_E$	$5\,T_h + 2\,T_E$
[44]	$T_h + T_f$	$3\,T_h$	$6\,T_h + T_f$	$6\,T_h + T_E$	$7\,T_h + 2\,T_f$	$9\,T_h + T_E$

8. Conclusions

We introduce a strong multi-factor authentication protocol to authenticate an administrator system—such as the owner of things in the IoT environment—by using a fuzzy extractor of the administrator's fingerprint, encrypted credential file, and application in the smart mobile device. We

applied our proposed protocol for AVISPA and the results indicate our protocol is safe against famous attacks such as MITM, replay, and insider. The real information of the administrator's system is saved in an anomalous way. The information of the administrator is saved as an encrypted credential file in a mobile device. In addition, in our work, the phase of changing the password in an online mode. Our proposed protocol has many security features such as mutual authentication, unlinkability, the anonymity of password and biometric, and a once-secure session key. Our protocol has the benefit to authenticate the administrator in the Internet system, owner's devices of IoT, and cloud computing. Finally, the work can be applied using modern environments such as cloud computing and cloud service provider. We can add another biometric fingerprint or a light factor such as a SMS message.

Author Contributions: Conceptualization, A.J.M. and A.A.Y.; Supervision, A.A.Y.; Writing—original draft, A.J.M.; and Writing—review and editing, A.A.Y.

Funding: This research received no external funding.

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

References

1. Baran, P. On Distributed Communications Networks. *IEEE Trans. Commun.* **1964**, *12*, 1–9. [CrossRef]
2. Licklider, J.C.R. *Memorandum for Members and Affiliates of the Intergalactic Computer Network*; Technical Report; Advanced Research Projects Agency: Washington, DC, USA, 1963.
3. Hsu, C.-L.; Lu, H.-P.; Hsu, H.-H. Adoption of the mobile Internet: An empirical study of multimedia message service (MMS). *Omega* **2007**, *35*, 715–726. [CrossRef]
4. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A Survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
5. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660. [CrossRef]
6. Mohsin, J.K.; Han, L.; Hegarty, R.; Hammoudeh, M. Two Factor vs Multi-factor, an Authentication Battle in Mobile Cloud Computing Environments. In Proceedings of the International Conference on Future Networks and Distributed Systems, Cambridge, UK, 19–20 July 2017.
7. Konoth, R.K.; van der Veen, V.; Bos, H. *How Anywhere Computing Just Killed Your Phone-Based Two-Factor Authentication*; Springer: Berlin/Heidelberg, Germany, 2016.
8. Kim, J.-J.; Hong, S.-P. A Method of Risk Assessment for Multi-Factor Authentication. *J. Inf. Process. Syst.* **2011**, *7*, 187–198. [CrossRef]
9. Wang, D.; Wang, P. Offline Dictionary Attack on Password Authentication Schemes Using Smart Cards. *Lect. Notes Comput. Sci.* **2015**, *7807*, 221–237.
10. Althobaiti, O.; Al-rodhaan, M.; Al-dhelaan, A. An Efficient Biometric Authentication Protocol for Wireless Sensor Networks. *Int. J. Distrib. Sens. Netw.* **2013**, *9*, 407971. [CrossRef]
11. Boneh, D. The Decision Diffie-Hellman Problem. In Proceedings of the International Algorithmic Number Theory Symposium, Portland, OR, USA, 21–25 June 1998.
12. Choi, Y.; Lee, Y.; Won, D. Security Improvement on Biometric Based Authentication Scheme for Wireless Sensor Networks Using Fuzzy Extraction. *Int. J. Distrib. Sens. Netw.* **2016**, *12*, 8572410. [CrossRef]
13. Park, Y.; Park, Y. Three-factor user authentication and key agreement using elliptic curve cryptosystem in wireless sensor networks. *Sensors* **2016**, *16*, 2123. [CrossRef]
14. Ndibanje, B.; Lee, H.-J.; Lee, S.-G. Security Analysis and Improvements of Authentication and Access Control in the Internet of Things. *Sensors* **2014**, *14*, 14786–14805. [CrossRef]
15. Sun, J.; Zhang, R. TouchIn: Sightless two-factor authentication on multi-touch mobile devices. In Proceedings of the 2014 IEEE Conference on Communications and Network Security, San Francisco, CA, USA, 29–31 October 2014; pp. 436–444.
16. Bruun, A.; Jensen, K.; Kristensen, D.; Nv, D.-R. Usability of Single- and Multi-Factor Authentication Methods on Tabletops: A Comparative Study. In Proceedings of the 5th IFIP WG 13.2 International Conference on Human-Centered Software Engineering, HCSE 2014, Paderborn, Germany, 16–18 September 2014; Springer: Berlin/Heidelberg, Germany, 2014.

17. Ometov, A.; Bezzateev, S.; Mäkitalo, N.; Andreev, S.; Mikkonen, T.; Koucheryavy, Y. Multi-Factor Authentication: A Survey. *Cryptography* **2018**, *2*, 1. [CrossRef]
18. Meixner, G. *Automotive User Interfaces*; Springer: Cham, Switzerland, 2017.
19. Hwang, M.-S.; Li, L.-H. A new remote user authentication scheme using smart cards. *IEEE Trans. Consum. Electron.* **2000**, *46*, 28–30. [CrossRef]
20. Khan, S.H.; Akbar, M.A.; Shahzad, F.; Farooq, M.; Khan, Z. Secure biometric template generation for multi-factor authentication. *Pattern Recognit.* **2015**, *48*, 458–472. [CrossRef]
21. Acharya, S.A.S. Two Factor Authentication Using Smartphone Generated One Time Password. *IOSR J. Comput. Eng.* **2013**, *11*, 85–90. [CrossRef]
22. Chen, X.; Tian, J.; Su, Q.; Yang, X.; Wang, F.-Y. A Secured Mobile Phone Based on Embedded Fingerprint Recognition Systems. In Proceedings of the IEEE International Conference on Intelligence and Security Informatics, ISI 2005, Atlanta, GA, USA, 19–20 May 2005; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3495, pp. 549–553.
23. Derawi, M.O.; Yang, B.; Busch, C. Fingerprint Recognition with Embedded Cameras on Mobile Phones. In Proceedings of the Third International ICST Conference on Security and Privacy in Mobile Information and Communication Systems, MobiSec 2011, Aalborg, Denmark, 17–19 May 2011; Springer: Berlin/Heidelberg, Germany, 2012; Volume 94, pp. 136–147.
24. Sin, S.W.; Zhou, R.; Li, D.; Isshiki, T.; Kunieda, H. Narrow Fingerprint Sensor Verification with Template Updating Technique. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2012**, *95*, 346–353. [CrossRef]
25. Ravi, H.; Sivanath, S.K. A novel method for touch-less finger print authentication. In Proceedings of the 2013 IEEE International Conference on Technologies for Homeland Security (HST), Waltham, MA, USA, 12–14 November 2013; pp. 147–153.
26. Dhillon, P.K.; Kalra, S. Secure multi-factor remote user authentication scheme for Internet of Things environments. *Int. J. Commun. Syst.* **2017**, *30*, e3323. [CrossRef]
27. Shrestha, B.; Tamrakar, S.; Mohamed, M.; Saxena, N. Theft-Resilient Mobile Wallets: Transparently Authenticating NFC Users with Tapping Gesture Biometrics. In Proceedings of the 32nd Annual Conference on Computer Security Applications, Los Angeles, CA, USA, 5–8 December 2016; pp. 265–276.
28. Buschek, D.; De Luca, A.; Alt, F. Improving Accuracy, Applicability and Usability of Keystroke Biometrics on Mobile Touchscreen Devices. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems—CHI '15, Seoul, Korea, 18–23 April 2015; pp. 1393–1402.
29. Buriro, A.; Crispo, B.; Del Frari, F.; Wrona, K. Touchstroke: Smartphone User Authentication Based on Touch-Typing Biometrics. In Proceedings of the ICIAP 2015 International Workshops on New Trends in Image Analysis and Processing, Genoa, Italy, 7–11 September 2015; Springer: Cham, Switzerland, 2015; Volume 9281, pp. 27–34.
30. Belk, M.; Fidas, C.; Germanakos, P.; Samaras, G. Computers in Human Behavior The interplay between humans, technology and user authentication: A cognitive processing perspective. *Comput. Human Behav.* **2017**, *76*, 184–200. [CrossRef]
31. Michelin, R.A.; Zorzo, A.F.; Campos, M.B.; Neu, C.V.; Orozco, A.M.S. Smartphone as a biometric service for web authentication. In Proceedings of the 11th International Conference for Internet Technology and Secured Transactions (ICITST), Barcelona, Spain, 5–7 December 2016; pp. 405–408.
32. Jeong, Y.; Park, J.S.; Park, J.H. An efficient authentication system of smart device using multi factors in mobile cloud service architecture. *Int. J. Commun. Syst.* **2015**, *28*, 659–674. [CrossRef]
33. Sun, J.; Zhong, Q.; Kou, L.; Wang, W.; Da, Q.; Lin, Y. A lightweight multi-factor mobile user authentication scheme. In Proceedings of the IEEE INFOCOM 2018—IEEE Conference on Computer Communications, Honolulu, HI, USA, 15–19 April 2018; pp. 831–836.
34. Whitfield, D.; Martin, E.H. New Directions in Cryptography. *IEEE Trans. Inf. Theory* **1976**, *22*, 644–654.
35. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *Inf. Secur. Appl.* **2004**, *3027*, 523–540.
36. Dodis, Y.; Ostrovsky, R.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and other Noisy Data. *SIAM J. Comput.* **2008**, *38*, 97–139. [CrossRef]
37. Sagar, F.A. Cryptographic Hashing Functions—MD5. Available online: http://cs.indstate.edu/~{}fsagar/doc/paper.pdf (accessed on 7 September 2019).

38. *Announcing the Advanced Encryption Standard (AES)*; Federal Information Processing Standards (FIPS); National Institute of Standards and Technology: Gaithersburg, MD, USA, 2001.
39. Pfitzmann, A.; Hansen, M. Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity Management—A Consolidated Proposal for Terminology. Available online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.153.6354 (accessed on 7 September 2019).
40. Maurya, A.K.; Sastry, V.N. Fuzzy Extractor and Elliptic Curve Based Efficient User Authentication Protocol for Wireless Sensor Networks and Internet of Things. *Information* **2017**, *8*, 136. [CrossRef]
41. The AVISPA Team. HLPSL Tutorial 2006. Available online: http://www.avispa-project.org/package/tutorial.pdf (accessed on 7 September 2019).
42. The AVISPA Team. AVISPA v1. 0 User Manual 2006. Available online: http://www.avispa-project.org/package/user-manual.pdf (accessed on 7 September 2019).
43. León, O.; Hernández-Serrano, J.; Soriano, M. A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor. *Int. J. Commun. Syst.* **2010**, *23*, 633–652.
44. Moon, J.; Lee, D.; Lee, Y.; Won, D. Improving Biometric-Based Authentication Schemes with Smart Card Revocation/Reissue for Wireless Sensor Networks. *Sensors* **2017**, *17*, 940. [CrossRef] [PubMed]