IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 17, 2019, accepted January 13, 2020, date of publication February 3, 2020, date of current version February 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2971024

Two-Factor Mutual Authentication Offloading for Mobile Cloud Computing

ABDELOUAHID DERHAB[1], MOHAMED BELAOUED[2], MOHAMED GUERROUMI[3], AND FARRUKH ASLAM KHAN[1], (Senior Member, IEEE)

[1]Center of Excellence in Information Assurance (COEIA), King Saud University, Riyadh 11415, Saudi Arabia
[2]LIRE Laboratory, Software Technologies and Information Systems Department, University of Constantine 2, Constantine 25001, Algeria
[3]Faculty of Electronic and Computer Science, USTHB University, Algiers 16111, Algeria

Corresponding author: Abdelouahid Derhab (abderhab@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research at King Saud University through Research Group under Grant RG-1439-021.

ABSTRACT Security analysts have shown that it is possible to compromise the mobile two-factor authentication applications that employ SMS-based authentication. In this paper, we consider that offloading mobile applications to the cloud, which is resource-rich and provides a more secure environment, represents a good solution when energy limitation and security constraints are raised. To this end, we propose an offloading architecture for the two-factor mutual authentication applications, and a novel two-factor mutual authentication scheme based on a novel mechanism, named virtual smart card. We also propose a decision-making process to offload the authentication application and its virtual smart card, based on three conditions: security, mobile device's residual energy, and energy cost. We analytically derive the lower-bound on the mobile application running time from the energy cost formula to perform offloading. We analyze and verify the security properties of the proposed architecture, and provide evaluation results of the two-factor mutual authentication protocol and the offloading decision-making process.

INDEX TERMS Decision-making, energy, offloading, security, two-factor authentication, virtual smart card.

I. INTRODUCTION

These days, mobile devices have become an essential part of our daily lives, due to the plethora of mobile applications that are capable to run different applications including social networking, gaming, and online banking. This is because they are provided with significant computing power and networking capabilities that compete with both laptops and desktop computers. Hence, complex applications, which are already used in the traditional desktop machines, are being migrated to mobile devices. However, two main issues have been raised by the research community: (1) mobile devices are equipped with limited battery capacity and these devices run different applications simultaneously, therefore, they run out of energy very quickly, (2) mobile devices are becoming the victims of cyber-criminals. For instance, Android OS has been highly targeted due its popularity [1]. A compromised mobile device, which is infected by a malware, can perform different malicious actions without the user's knowledge and consent including accessing sensitive information such as bank account and GPS location, as well as performing SMS premium rate fraud activities.

To deal with the above two issues, we consider that offloading mobile applications to the cloud, which is resource-rich and can provide a more secure environment, presents a good solution when energy limitation and security constraints are raised. In fact, mobile offloading is a computation paradigm, which has been considered as a solution for mobile devices with limited resources with respect to processing, battery, and storage, and it allows executing heavyweight and resource-consuming tasks on the cloud instead of the mobile devices [2]–[6]. In addition, as the mobile operating system offers limited security, full or partial offloading allows running security applications in a more secure environment [7]–[11].

Nowadays, most of the mobile two-factor authentication applications employ SMS-based authentication, i.e., to access an account, the users are required to provide something they know (password) and something they have (one-time verification code sent to the mobile device). However, it has been shown that it is possible to bypass this security mechanism. In 2016, an Android malware targeted more than twenty mobile banking applications, and succeeded in stealing the

The associate editor coordinating the review of this manuscript and approving it for publication was Raja Wasim Ahmad.

login credentials of the user and the SMS verification code, which is sent by the authentication server [12], [13]. When a user launches the legitimate mobile banking application, a fake login interface is triggered by the malware to cover the original banking application. The login credentials, which are filled by the user in the fake application, are sent to the attacker. The malware can also capture the SMS verification code sent to the user, and then it can send it to the attacker. This attack succeeds in tricking the user by employing a visual phishing technique, i.e., creating a fake screen that is visually similar to the legitimate application.

To tackle the above-mentioned threat, we propose a novel two-factor mutual authentication scheme, which replaces the SMS verification code with a novel concept called *virtual smart card*. The two-factor mutual authentication schemes, based on password and smart card, have been widely investigated in the literature [14]–[29]. In such schemes, the user provides his ID and password, and inserts the smart card into the smart card reader. If the authentication succeeds, the user accesses his account on the remote server. This mechanism has shown to be effective in providing secure remote access. We design a virtual smart card, which leverages the security features of Android OS to allocate a secure storage (i.e., virtual smart card) to a specific two-factor mutual authentication application.

We also propose offloading the two-factor mutual authentication application to a more secure environment (i.e., cloud). To this end, we propose a decision making process that offloads the authentication application according to three conditions: security, mobile device's residual energy, and energy cost.

The main contributions of the paper are the following:

- We propose a novel two-factor mutual authentication scheme based on a novel mechanism, named virtual smart card.
- We propose an offloading architecture for the two-factor mutual authentication application.
- We present a decision-making process to offload the authentication application and its virtual smart card, based on three conditions: security, mobile device's residual energy, and energy cost.
- We analytically derive from the energy cost formula, the lower-bound on the application running time on the mobile device, denoted by min t_m, to perform offloading. The formula considers different heterogeneous parameters including network bandwidth, wireless communication model, energy dissipation of other applications running on the mobile device, and the speed execution factor between the cloud server and the mobile device.
- We provide security analysis and formally verify the security properties of the two-factor mutual authentication scheme using BAN Logic and ProVerif techniques.
- We present performance evaluation of the two-factor mutual authentication scheme and the offloading decision-making process.

The remainder of this paper is structured as follows: In section II, we present the offloading architecture for the two-factor mutual authentication application. Section III describes the offloading decision-making process. In Section IV, we describe the proposed two-factor mutual authentication scheme. Security analysis and formal verification are provided in Section V and Section VI respectively. Evaluation results of the two-factor mutual authentication scheme and the offloading decision-making process are shown in Section VII. Finally, Section VIII concludes the paper.

II. OFFLOADING ARCHITECTURE FOR THE TWO-FACTOR MUTUAL AUTHENTICATION APPLICATION

In this section, we present the attack and security models, as well as the offloading architecture of the proposed two-factor mutual authentication protocol.

A. ATTACK MODEL

We consider the following attacks:

- *Visual phishing*: a malicious application, which is visually similar to the authentication application, is installed on the mobile device.
- The authentication mechanism between the user and the server might be targeted by the following threats: compromise of user anonymity, user traceability, password guessing attacks (online and offline), stolen-verifier attack, privileged insider attack, replay attack, modification attack, user and server impersonation attacks, compromise of mutual authentication, compromise of perfect forward secrecy, compromise of password chosen and update, and compromise of virtual smart card revocation, as will be explained in Section V.

B. SECURITY MODEL

We make the following assumptions on security of the mobile device:

- The mobile device runs on Android OS, and Android kernel is initially free from malicious codes.
- We employ the Sandboxing mechanism that is implemented in the Android kernel. This mechanism allows separating applications from each another. Android assigns to each application a unique user identifier, which is considered as a separate process. Also, one application (resp., user) is not allowed to access or change any file belonging to another application.
- The privilege escalation attack cannot take place. If a malware succeeds to access the root account, it will be able to bypass the sandboxing security mechanism, and thus totally control the operating system. This attack can be thwarted by adopting some solutions [30], or by using SELinux-enabled Android kernel. In SELinux, data are protected by applying a custom security policy, which specifies the list of applications that can access the data [31]–[34].

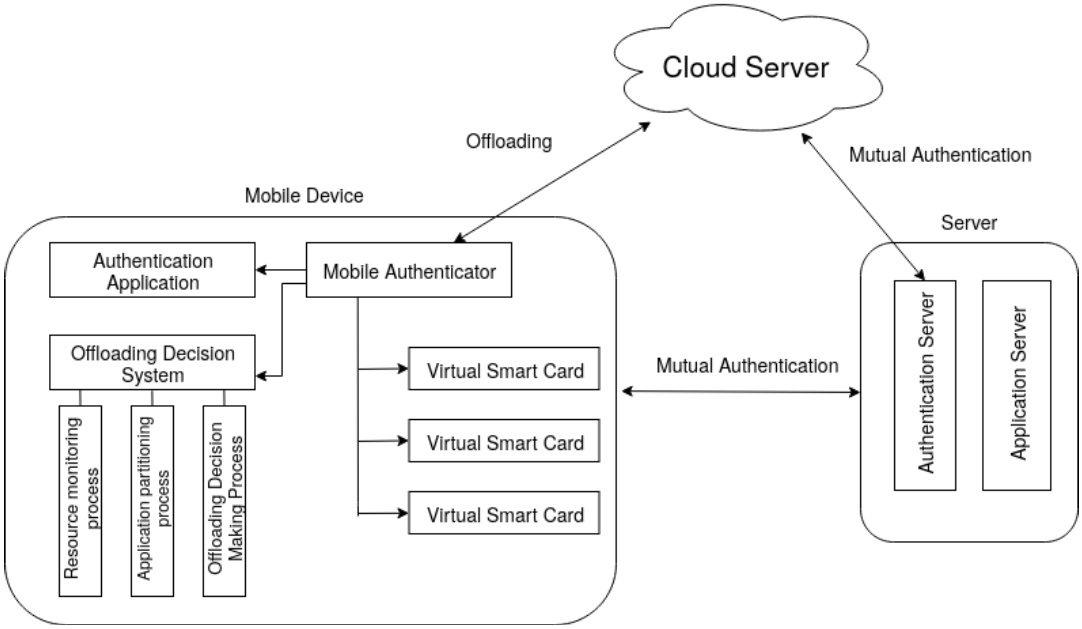


FIGURE 1. Offloading architecture.

C. ARCHITECTURE OVERVIEW

As shown in Figure 1, we present an overview of the offloading architecture that offloads the mobile two-factor mutual authentication application and its associated data to the cloud. The architecture comprises the following elements:

- *Mobile device*: It runs the client authentication application, which permits users to send their login credentials to an authentication server. If the authentication succeeds, the user is granted access to the remote application server. The mobile device also runs the following components:
  - *Mobile authenticator*: It comprises the list of authentication applications, which are selected by the user and installed on the mobile device. It applies a policy enforcement to manage the access to the virtual smart cards. Each virtual smart card is only accessible by its corresponding authentication application.
  - *Virtual smart card*: It is a secure storage space that is allocated for each client authentication application. It contains all the credentials that exist in the conventional smart card.
  - *Mobile offloading system*: It is responsible for offloading the authentication application and its associated virtual smart card to the cloud server. It is composed of three processes: (1) Resource monitoring process, (2) Application partitioning process, and (3) offloading decision-making process.
- *Server*: It hosts the authentication server that validates and authenticates remote users aiming to access an application server.
- *Cloud server*: It is used to run the offloaded authentication application.

D. MOBILE OFFLOADING SYSTEM

The offloading system consists of three main processes:

- *Resource monitoring process*: It collects information about the mobile device's resources such as battery information (i.e., global battery dissipation, battery dissipation of each application, and residual battery level), CPU information (i.e., CPU utilization ratio, CPU speed), and network bandwidth.
- *Application partitioning process*: It cuts the application into two partitions: (a) local partition, which is executed by the mobile device. It represents, in our case, the graphical user interface (GUI) of the authentication application, and (b) remote partition, which is offloaded to the cloud, and represents the mutual authentication scheme (Section IV).
- *Offloading decision-making process*: It is invoked by the mobile authenticator, and decides if it is worth offloading the authentication application and its corresponding virtual smart card to the cloud.

E. OFFLOADING OPERATIONS

- When the user launches an authentication application, and enters his login credentials, the mobile authenticator sends a message to the user, and requests him to provide a password to acess its virtual smart card. This operation simulates the insertion of the smart card in the card reader.
- When the user provides the required password, the mobile authenticator, on behalf of the application, triggers the offloading decision making process.
  - If the process decides not to perform offloading, then the authentication application interacts with the authentication server, and establishes a secure

TABLE 1. Offloading parameters.

Parameter	Description
t_m	Running time on the mobile device (sec)
t_s	Running time on the server (sec)
E_m	Residual energy of the mobile device (Joule)
S_m^{Auth}	Energy dissipation rate (Joule/sec) of the authentication application.
S_m^i	Energy dissipation rate (Joule/sec) of application i running on the mobile device
$P_{m,T}$	Energy rate to send data
$P_{m,R}$	Energy rate to receive data
D	Size of the offloaded data
$TR_{m,B}$	The transmission rate between the mobile device the base station.
$T_{m,T}$	Data transmission time
$T_{m,RT}$	Duration of the reception mode
$E_{m,B}$	Energy consumption to transmit data from the mobile device to the mobile base station
W_m	Mobile transmission Bandwidth
K	Constant path loss factor
P_m	Transmission power of mobile device
N_0	Noise Power Spectrum Density
$d_{m,B}$	Distance between the mobile device and the mobile base station
γ	Path loss exponent

connection between the application server and the mobile device.

- If the process decides to perform offloading, then the mobile offloading system is invoked to offload authentication application without the graphical interface, and its corresponding virtual smart card. They are transmitted to the cloud server using the server's public key. The offloaded application interacts with the authentication server, and establishes a secure connection, which is forwarded to the mobile device to access its account on the application server. When the offloading process terminates, the offloaded data are deleted from the cloud server.

III. OFFLOADING DECISION-MAKING PROCESS

The decision to offload the authentication application and its associated virtual smart card to the cloud is performed in the following order according to three conditions:

- *Security condition*: It occurs when the graphical user interface (GUI) of the authentication application is visually similar to any running application. This verification operation is performed, as explained in [35].
- *Residual energy condition*: It occurs when the residual lifetime of the mobile device is less than t_m. Formally: $\frac{E_m}{S_m^{Auth}} < t_m$.
- *Energy cost condition*: If the security and the residual energy conditions are not met, we verify the energy cost condition, which determines if it is worth offloading the application from the mobile device to the cloud, i.e., the energy cost of the mobile device without performing offloading is higher than that with offloading.

In the following, we define the equations related to the energy cost condition. The parameters listed below are summarized in Table 1. We first present the radio energy consumption model that is used in [36]. The mobile device can be in one of the following modes: Transmission, Reception, Idle, or Sleep. Nonetheless, we use a simple energy consumption model, which only considers the communication phase. The device can either be in the transmission or the reception mode. The radio energy consumption model between the mobile device and the mobile base station is given by:

$$E_{m,B} = P_{m,T} \times T_{m,T} + P_{m,R} \times T_{m,R} \tag{1}$$

It means that after transmitting the data, the mobile device switches to the reception mode for a duration $T_{m,R} = T_{m,T}$, which is the time to transmit D bits to the base station, and is expressed by the following equations:

$$T_{m,T} = \frac{D}{TR_{m,B}} \tag{2}$$

$$TR_{m,B} = W_m \log_2\left(\frac{KP_m}{N_0 W_m d_{m,B}^{\gamma}}\right) \tag{3}$$

The energy consumption $E_{m,B}$ can be expressed as follows:

$$E_{m,B} = \frac{(P_{m,T} + P_{m,R})D}{TR_{m,B}} \tag{4}$$

To meet the energy cost condition, the following inequality must hold true:

$$S_m^{Auth} \times t_m + \sum_{apps\ i} S_m^i \times t_m > \sum_{apps\ i} S_m^i \times (t_s + 2T_{m,T}) + E_{m,B} \tag{5}$$

It follows that:

$$S_m^{Auth} \times t_m + \sum_{apps\ i} S_m^i \times t_m > \sum_{apps\ i} S_m^i \times \left(t_s + 2\frac{D}{TR_{m,B}}\right) + \frac{(P_{m,T} + P_{m,R})D}{TR_{m,B}} \tag{6}$$

Let $t_m = \Delta t_s$, where $\Delta > 1$ denotes the speed execution factor between the cloud and the mobile device. The above equation becomes:

$$\left(S_m^{Auth} + \sum_{apps\ i} S_m^i\left(1 - \frac{1}{\Delta}\right)\right) t_m > \left(2 \sum_{apps\ i} S_m^i + P_{m,T} + P_{m,R}\right) \times \left(\frac{D}{TR_{m,B}}\right) \tag{7}$$

By replacing $\sum_{apps\ i} S_m^i$ with β, we get:

$$t_m > \frac{(2\beta + P_{m,T} + P_{m,R}) \times (\frac{D}{TR_{m,B}})}{(S_m^{Auth} + \beta(1 - \frac{1}{\Delta}))} \quad (8)$$

IV. PROPOSED TWO-FACTOR MUTUAL AUTHENTICATION SCHEME

A. PRELIMINARIES

The proposed scheme is composed of five phases, namely: registration, login, authentication, password change, and revocation.

The notations, which are used to describe the authentication scheme, are shown in Table 2.

TABLE 2. Notations.

Notation	Description
U_i	The user
ID_i	Identity of user U_i
PW_i	Password of user U_i
S	The remote application server
x	The secret key of server S
e	The public key of server' S
N	Number of registrations with S by user U_i
T_i	The user's timestamp
T_s	The server's timestamp
$E_Q(m)$	Encryption of message m using $key\ Q$
$D_Q(m)$	Decryption of message m using $key\ Q$
Sk	Session Key
$h(.)$	Secure one-way hash function
$\otimes$	Bitwise XOR operation
$\|\|$	Cconcatenation operation

1) REGISTRATION PHASE

In this phase, the mobile user through the authentication application registers with the remote server. The user U_i generates his/her ID_i and password PW_i and only sends ID_i to the server S. The latter checks the validity of ID_i. If it is found legitimate, the server S generates a virtual card number SC_i and saves it in its database:

$(ID_i, RID_i) = (ID_i, E_x(SC_i, ID_i, N)$, where N denotes the number of registrations performed by user U_i. If $N = 0$, it represents the first registration. Otherwise, the value of N is set to $(N + 1)$). Then, the server computes the following:

- It generates a random number b, and computes $AID_i = E_x(ID_i \,||\, b)$.
- $J = h(x \,||\, ID_i \,||\, N \,||\, SC_i)$, where J is the common data shared between the mobile device and the server, and is used to check the validity of the entered data.

After that, the server stores $\{AID_i, SC_i, N, J, n, e, h(.), E_Q(.), D_Q(.)\}$ in its database and sends it to the authentication application through a secure channel.

After receiving the above information, the mobile authenticator is invoked to store them in a new secure storage space, called the virtual smart card. Then:

- It generates a random number r, and computes $RPW_i = h(PW_i \,||\, r)$.
- $L = J \otimes RPW_i$, and replaces J by L in the virtual smart card of the mobile device.

Figure 2 summarizes the message exchanges between the mobile device and the server during the registration phase.

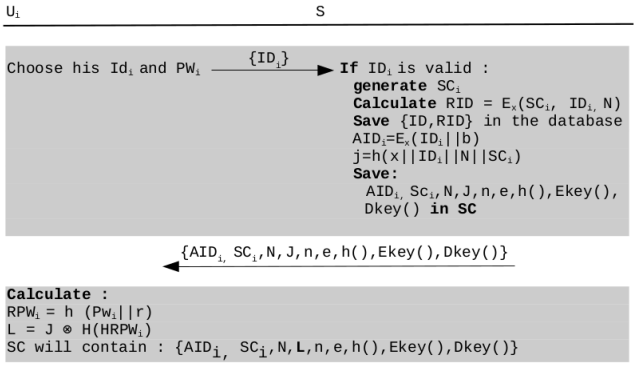


FIGURE 2. Registration phase.

2) LOGIN PHASE

When a legitimate user U_i wants to login to the authentication server, he first needs to enter his identity ID_i and password PW_i through the authentication application, and provides the password of the virtual smart card. Then, it retrieves the random number r and computes:

- $RPW_i = h(PW_i||r)$
- $J = L \otimes h(RPW_i)$ (J is computed using the password provided by the user when he/she requests a new connection).
- $F = J^a \ mod\ n$
- $C_1 = h(T_i||J||F)$
- $D = C_1^e \ mod\ n$
- $IZ_i = E_{C_1}(AID_i, N, SC_i, F)$

The authentication application sends the message $m = \{IZ_i, T_i, D\}$ to the authentication server.

3) AUTHENTICATION PHASE

Upon receipt of the login message m, the authentication server checks if $T_i - T' > \Delta T$, such that:

T' : is the current timestamp.

ΔT : is the valid time interval.

If the above inequality holds true, the login request is rejected. Otherwise, the server performs the following operations:

- It decrypts D with its secret key x to obtain C_1 using: $C_1 = D^x \ mod\ n$.
- It decrypts IZ_i with C_1 to retrieve the values of AID_i, N, SC_i, F using: $D_{C_1}(IZ_i) = (AID_i, N, SC_i, F)$.
- It decrypts AID_i using: $D_x(AID_i) = (ID_i, b)$ to obtain ID_i
- It decrypts RID_i to obtain N, SC_i, ID_i.

Finally, it checks if ID_i already exists in the database. If so, it compares the information of the decrypted virtual smart card number with the one stored in the database along with

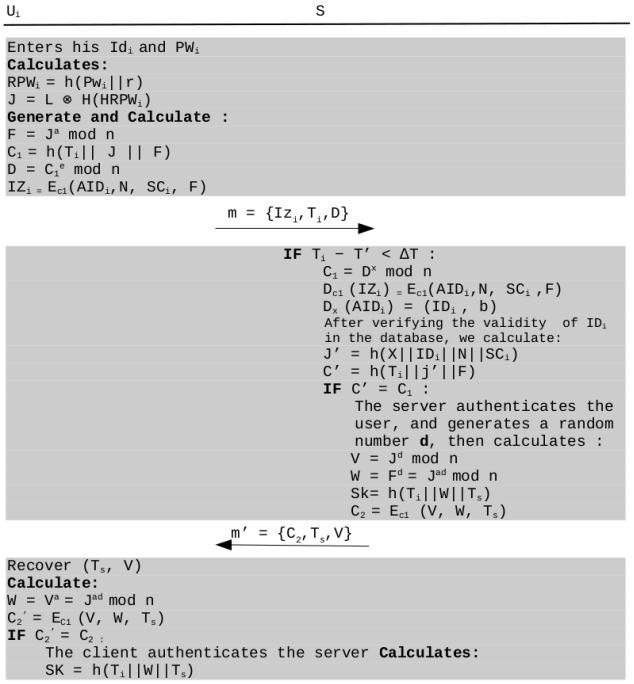


FIGURE 3. Login and authentication phases.

verification of its validity (if the card number does not refer to a stolen or frozen virtual smart card). Then it computes:

- $J' = h(x||ID_i||N||SC_i)$
- $C_1' = h(T_i||J'||F)$

If $C_1' = C_1$, the user is authenticated, and a random number d is generated by the server, which computes:

- $V = J^d \ mod \ n$
- $W = F^d = J^{ad} \ mod \ n$
- $Sk = h(T_i||W||T_s)$
- $C_2 = E_{C_1}(V, W, T_s)$

Finally, it sends the message $m' = \{C_2, V, T_s\}$ to the user, where T_s corresponds to the timestamp when the server sent the message.

When the user receives the message m' at time T'', he checks if $T_s - T'' < \Delta T$. If the inequality holds true, it computes:

- $W = V^a = J^{ad} \ mod \ n.$
- $C_2' = E_{C_1}(V, W, T_s)$

If $C_2 = C_2'$, the server S is authenticated, and the user U_i calculates his session key $Sk = h(T_i||W||T_s)$.

Figure 3 summarizes the message exchanges between the user and the server during the login and authentication phases.

4) PASSWORD CHANGE PHASE

When a user wants to update his password, he provides his old password PW_i and selects a new password PW_i'. The mobile authenticator computes: $RPW_i' = h(PW_i'||r)$ and $L' = L||h(RPW_i)||h(RPW_i')$. Then, L is replaced with L' in the virtual smart card.

5) LOST VIRTUAL SMART CARD REVOCATION PHASE

When U_i loses his mobile device, its virtual smart card is lost as well. The user sends a request to S to generate a new virtual smart card. After verifying that the user's national identity card is still valid, the revocation process is the same as the registration phase, except for replacing N with $N+1$, and replacing $RID_i = E_x(ID_i, N, SC_i)$ with $RID_i = E_x(ID_i, N, SC_i')$.

V. SECURITY ANALYSIS

The proposed two-factor mutual authentication protocol can resist against the following attacks:

A. USER ANONYMITY

By user anonymity, we mean that U_i is only known by the server S and the user himself. As ID_i is hidden in AID_i, RID_i and IZ_i, the attacker needs to decrypt AID_i and RID. The private key x is only known by the server. Thus, the attacker cannot identify the user who established the authentication session with the server. Therefore, the proposed scheme ensures anonymity.

B. USER UNTRACEABILITY

This property means that an attacker cannot know if two or more executions of the schemes are related to the same user. In the proposed scheme, each element of the login message $m = \{IZ_i, T_i, D\}$ is dynamic for each session, as the computation of IZ_i and D involve random nonces a and b that change from one session to another. In the registration phase, the identity of the user is encrypted as AID_i using the secret key x, and sent to the user in a secure way. In the login phase, AID_i is encrypted as IZ_i using the key $C1$, which is calculated using a random variable a. Therefore, each ID_i is encrypted differently for each session, even the same secret key x is used. Since the attacker has no knowledge of the secret key x, he will not be able to tell if two different values of AID_i correspond to the same ID_i. Also, as there is a different value of $C1$ for each session, which is associated with the timestamp T_i, the attacker is not able to tell if two executions of the scheme are related to the same user. Thus, user untraceability property is satisfied.

C. ONLINE PASSWORD GUESSING

In this attack, an unauthorized user tries several passwords to get the correct one. Any user that wants to login to the server, has to enter the right login message $m = \{IZ_i, T_i, D\}$. RPW_i is produced from PW_i and the random number r, and encrypted as IZ_i using the key $C1$. As $C1$ changes for each session, it will be difficult for an attacker to produce the correct password. In addition, the login depends on C_1, so its value is different for each session. Thus, the server can detect this attack if the number of failed login attempts is limited.

In our scheme, and in order to produce the login message, it is necessary to enter a correct password. If we assume

that an attacker can guess the correct password, the server can detect this attack (i.e., password guessing) by checking whether $C_1 = C_1'$ or not.

D. OFFLINE PASSWORD GUESSING

If the attacker can get the stored information in the virtual smart card, he cannot guess the password PW_i as it is calculated by a random number r and protected by a hash function. Then, it is encrypted using a random number a. Also, it is not possible for an attacker to guess the password from the authentication message m, or the server's response, as they do not contain any password-related information.

E. PRIVILEGED INSIDER ATTACK

A user can be impersonated by the server's system administrator if the latter knows the password of the user. In this scheme, the server does not keep any password table. In addition, a random number r as well as a hash function protect the password during the registration and the login phases, which makes the proposed scheme resilient against this attack.

F. STOLEN-VERIFIER ATTACK

In this attack, an attacker steals or modifies verification data that are stored by the server (e.g., plain-text, hashed passwords, etc.). In this scheme, the authentication information $J = h(x||ID_i||N||SC_i)$ can only be verified by the server as it has the secret key x. Also, there is no way to derive password or verification information from PW_i, which is stored in the server. Therefore, this property is satisfied.

G. MODIFICATION ATTACK

An attacker can transmit a captured and modified message rather than the original one. In this scheme, the verification data are: (a) C_1 that is concatenated with J and F, and then hashed, and (b) C_2 that is encrypted using C_1. Any modification is not possible because an attacker cannot generate C_1 and C_2 without the knowledge of J. Thus, this scheme ensures security against this attack.

H. REPLAY ATTACK

An attacker replays the eavesdropped messages, such as the login message $m = IZ_i, T_i, D$ or the server's response message $m' = C_2, T_s, V$. For each message, the validity period is limited by a timestamp, and we can verify this attack by checking if the timestamp is valid or not. Thus, the scheme is resilient against the replay attack.

I. USER IMPERSONATION ATTACK

To impersonate a legitimate user U_i, an attacker must then calculate $C_1 = h(T_i||J||F)$ to prove his identity. However, the attacker cannot calculate the correct value of C_1 since he does not have the secret key x or $J = h(x||ID_i||N||SC_i)$, which is obtained from L and RPW_i. Thus, this scheme is resilient against this attack.

J. SERVER IMPERSONATION ATTACK

If an attacker wants to impersonate the server, he needs to produce a valid response $C_2 = E_{C1}(V, W, Ts)$. However, the attacker does not know C_1, that is computed from J, which is only stored at the server. Therefore, he cannot calculate C_2 and generate a valid response. Thus, the scheme can resist against the server impersonation attack.

K. MUTUAL AUTHENTICATION

This property means that both the server and the user are able to verify each other's identity. It has been shown above that only the server that has the correct secret key can pass verification at the user level using C_2. Also, only legitimate users with correct passwords can pass the verification at the server level using C_1. Hence, mutual authentication is ensured.

L. PERFECT FORWARD SECRECY

When the user successfully logins to the server, it is important to ensure the integrity and confidentiality of the transmitted data against exposure, modification, and deletion.

In this scheme, after performing mutual authentication, the server and the user U_i calculate the session key $Sk = h(T_i||W|T_s)$, where only the user and the server know W. Furthermore, the session key Sk is computed separately, and is distinct for each login session as the user and the server generate a and d respectively, which are random values and are different for each session. An attacker cannot deduce W by only knowing F and V. Hence, if we assume that some session keys are obtained, it is not possible to generate previous or future session keys.

M. PASSWORD CHOSEN AND UPDATE

In this scheme, the passwords can be freely changed by the user.

N. VIRTUAL SMART CARD REVOCATION

In this scheme, a user can request revocation of the virtual smart card. The server increases N by 1, and issues a new card with a new number to the user. As N is secured through a hash function, an attacker cannot use old virtual smart cards to login to the server.

O. COMPARISON OF SECURITY PROPERTIES

In Table 3, we compare the proposed scheme with other authentication schemes [20]–[26], [28], [29], [37]–[45] with respect to the above-mentioned security properties. Based on some comparative studies [14], [46], we assign three symbols to each security property, as follows:

- ✓: The scheme satisfies the security property.
- ✗: The scheme does not satisfy the security property.
- - : There is no available information on the scheme's property satisfaction.

From Table 3, we can observe that our scheme can achieve better security, as it satisfies 14 satisfied security properties,

TABLE 3. Security properties of two-factor authentication schemes.

Scheme	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14
Khan et al. [20]	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Chen et al. [21]	✗	✗	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
Jiang et al. [22]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
Wu et al. [23]	-	✗	✓	✗	✗	✗	✓	✗	✓	-	-	-	✗	✗
Debiao et al. [24]	✗	-	✓	✗	✓	✗	✓	✓	✗	-	-	-	✓	✗
Wei et al. [25]	✗	-	✓	✗	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓
Lee et al. [26]	-	-	✗	✓	✓	✓	✗	✗	✓	-	-	✓	✗	✓
Farash et al. [28]	✗	-	-	✗	✓	✗	-	✓	✗	-	-	-	✓	✗
Wen et al. [29]	✗	✓	-	✗	✓	-	-	✓	✓	-	-	✗	✓	✗
Mishra et al. [37]	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✓	✓	✗
kumari et al. [38]	✓	✓		✓	✓	✓	-	✓	✓	-	✓		✓	✗
Zhu et al. [39]	✗	✗	✓	✓	✓	✓	✗	✓	✓	-	✓	✓	✗	✓
Das et al. [40]	✗	-	✓	✓	✓	✓	✗	✓	✓	-	-	-	✓	✓
Islam et al. [41]	✓	-	✗	✓	✓	✗	✗	✓	✗	✗	✗	✓	✓	✗
Chaudhry et al. [42]	✓	✓	✗	✓	✓	✓	✗	✓	✗	✗	✓	✓	✓	-
Jiang et al. [43]	✓	✓	✗	✓	✓	✓	✓	✓	✓	-	-	✓	✗	✓
Tu et al. [44]	✗	-	-	✓	✓	✓	-	✗	✗	-	✓	✓	-	✓
Wen et al. [45]	✗	-	-	✗	✗	-	-	✓	✓	-	-	✗	-	✓
Our scheme	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

P1: Ensure user anonymity, P2: Ensure user untraceability, P3: Resist online password guessing attack, P4: Resist off-line password guessing attack, P5: Resist privileged insider attack, P6: Resist stolen-verifier attack, P7: Resist modification attack, P8: Resist replay attack, P9: Resist user impersonation attack, P10: Resist server impersonation attack, P11: Ensure mutual authentication, P12: Ensure perfect forward secrecy, P13: Ensure password change, P14: Ensure smart card revocation

and its closest competitor is Jiang et al. scheme [22], which provides 13 satisfied properties.

VI. FORMAL VERIFICATION

The purpose of the verification is to test the robustness of the security protocols. In general, we verify whether the security protocol meets the expected security requirement or not. In this section, we employ two formal techniques: BAN Logic [47] and ProVerif [48].

A. FORMAL VERIFICATION USING BAN LOGIC

According to BAN Logic, we determine the goals to be verified of our proposed authentication scheme:

Goal 1: $U_i \mid\equiv (U_i \overset{Sk}{\leftrightarrow} S)$

Goal 2: $S_i \mid\equiv (U_i \overset{Sk}{\leftrightarrow} S)$

The proposed authentication scheme can be written in the following form:

Message 1: $S : \{AID_i, N, SC_i, F, D, T_i\}C1$

Message 2: $U_i : \{C_2, T_s, V\}C1$

We make the following assumptions about the initial status of the scheme:

A.1 : $U_i \mid\equiv J^a \ mod \ n$

A.2 : $S \mid\equiv J^d \ mod \ n$

A.3 : $U_i \mid\equiv (U_i \overset{C1}{\leftrightarrow} S)$

A.4 : $S \mid\equiv (U_i \overset{C1}{\leftrightarrow} S)$

A.5 : $U_i \mid\equiv S \Rightarrow J^d \ mod \ n$

A.6 : $S \mid\equiv U_i \Rightarrow J^a \ mod \ n$

Based on the above assumptions, we analyze the form of the scheme and the main evidence procedures:

According to Message 1, we obtain : $S \leftarrow \{J^a \ mod \ n\}C1$

According to Assumption A.4 : $(U_i \overset{C1}{\leftrightarrow} S)$

According to the message-meaning rule, we get:

$$S \mid\equiv U_i \rightleftharpoons J^a \ mod \ n \tag{9}$$

According to Assumption A.2 and in accordance with the freshness-conjuncatenation rule, we obtain the clause:

$$S \mid\equiv J^a \ mod \ n \tag{10}$$

By Statement 9 and Statement 10, and according to the nonce-verification rule, we obtain:

$$S \mid\equiv U_i \mid\equiv J^a \ mod \ n \tag{11}$$

By Statement 11 and Assumption A.6, and according to the jurisdiction rule, we obtain:

$$S \mid\equiv J^a \ mod \ n \tag{12}$$

By Statement 12 and Assumption A.2, we obtain:

$$S \mid\equiv J^{ad} \ mod \ n \tag{13}$$

In accordance with $Sk = h(T_i||W||T_s)$, we conclude: $S_i \mid\equiv (U_i \overset{Sk}{\leftrightarrow} S)$ {Goal 2}

From Message 2, we deduce: $S \leftarrow \{J^d \ mod \ n\}C1$

According to Assumption A.3: $U_i \mid\equiv (U_i \overset{C1}{\leftrightarrow} S)$

According to the message-meaning rule we get:

$$U_i \mid\equiv S \rightleftharpoons J^d \ mod \ n \tag{14}$$

According to Assumption A.1 and in accordance with the freshness-conjuncatenation rule, we obtain the clause:

$$U_i \mid\equiv J^d \ mod \ n \tag{15}$$

By Statement 14 and Statement 15, and according to the nonce-verification rule, we obtain:

$$U_i \mid\equiv S \mid\equiv J^a \ mod \ n \tag{16}$$

By Statement 16 and Assumption A.5, and according to the jurisdiction rule, we obtain:

$$U_i \mid\equiv J^d \ mod \ n \tag{17}$$

By Statement 17 and Assumption A.1, we obtain:

$$S \mid\equiv J^{ad} \ mod \ n \tag{18}$$

In accordance with : $Sk = h(T_i||W||T_s)$, we conclude:
$U_i \mid\equiv (U_i \underset{\text{Sk}}{\leftrightarrow} S)\{Goal1\}$

B. FORMAL VERIFICATION USING PROVERIF

1) BACKGROUND ON PROVERIF

ProVerif [48] is a tool that performs automatic verification of cryptographic protocols [49]. Compared to other existing verification tools [50], [51], it has many features and high expressiveness.

ProVerif is executed automatically, and errors can be easily detected. It checks secrecy and authentication properties using different cryptographic primitives, for instance, symmetric and asymmetric encryption functions, hash functions, digital signatures, etc.

2) PROPERTIES TO BE CHECKED

The properties to be checked are as follows:

1) **Secrecy properties:** This property [52] aims to ensure the transmission of a message without being visible to certain agent(s) (in general, the intruder), and includes the following:
   - **Message secrecy:** A message is said to be secret if the intruder does not know about it at the end of the scheme execution.
   - **Message confidentiality:** The confidentiality of a message is a local secret that is restricted to a single possible recipient. So the communication $S!U : m$ is said to be confidential if S is certain that only U will receive message m; i.e., S can ensure that no one intercepts the message m and that the recipient is U.
2) **Authentication properties:** It is possible to define authentication properties for both messages and agents. The objective is that an agent can confirm the origin of the message it receives.
   - **Authentication of a message:** A message m is said to be authenticated by B in communication $A!B$: if B can ensure that message m it receives has been generated by A. It is also possible to check the integrity of m. Indeed, this property ensures that the message sent by A has not been modified until it is received by B.
   - **Authentication of an agent:** It verifies that no intruders spoof A's identity during the scheme execution [52].

3) VERIFICATION RESULTS

The authentication scheme is modelled as a simultaneous execution of two processes: the mobile device and the server. Formally, it will be written as: (process: $!U|S$). Each process

```
(* -------------channel---------------------*)

free sch: channel.

(* -------------shared keys ---------------------*)
event UserAuthed(bitstring).
event UserStarted(bitstring).

(*---------------- constants and variables-----------------*)
free sk': bitstring [private]. ** session key **
free sk: bitstring [private].
const ID: bitstring.
const Email: bitstring.
const PW: bitstring [private].
const PW': bitstring [private].
const p: bitstring.
free q: bitstring [private].
free one: bitstring [private].

(*---------------constructor----------------*)
fun h(bitstring): bitstring. ** hash function **
fun senc(bitstring, bitstring): bitstring.** Symetric Encryption **
fun xor(bitstring, bitstring): bitstring.
fun exp(bitstring, bitstring): bitstring.
fun mult(bitstring, bitstring): bitstring.
fun mod(bitstring, bitstring): bitstring.
fun sub(bitstring, bitstring): bitstring.

(*---------------destructors & equations----------------*)
reduc forall x: bitstring, y: bitstring; sdec(senc(x,y),y)=x.
equation forall x: bitstring, y: bitstring; xor(xor(x, y),y)=x.
```

FIGURE 4. The proverif declaration part.

```
Query attacker (sk).
Query attacker (sk').
Query id:bitstring;inj-event(UserAuthed(id))==>inj-event(UserStarted(id))
```

FIGURE 5. ProVerif requests for attack tests.

```
let S=
in (sch, ID1 : bitstring) ;
let n = mult(p,q) in
new X: bitsring;
new e: bitsring;
if mod(mult(e,X),mult((sub(p,one)),sub(q,one)))=one then
(
new b:bitsring;
new N:bitsring;
new SCI:bitstring;
let j=h(((X,ID1,N,SCI))) in
let RID=senc(((ID1,N,SCI)),X) in
let AID=senc((ID1,b),X) in

out (sch,(AID,SCI,N,J,n,e));
in (sch,(IZ':bitsring,T1':bitsring,D':bitstring));

let (C1':bitstring)=mod(exp(D',X),n) in
let (AID3:bitsring,N3:bitsring,SCI3:bitsring,C1':bitstring,F':bitsring)
=sdec(IZ',C1') in
let (ID3:bitsring,b1:bitsring)=sdec(AID3,X) in

event UserAuthed (ID3);
let J1=h((((X,ID3,N3,SCI3)))) in
let C1''=h(((T1',J1,F'))) in

if C1'=C1'' then
new d: bitsring;
new T2: bitsring;
let V=mod(exp(J1,d),n) in
let W=exp(F',d) in
let C2=senc(((T2, V, W)), C1') in

out (sch,(C2,T2,V));
let sk=h(T1',W, T2) in
0
).
```

FIGURE 6. The source code of the server agent.

defines the behavior of a single participant in the applied pi calculus. The declaration part is shown in Figure 4. The part related to the verification requests: the authentication and the secrecy of the session key of the scheme, is modelled in Figure 5. The server process illustrates the behavior of S during the authentication phase. Upon receipt of message (IZ_i, T_i, D) from U, S decrypts D and IZ_i to obtain C_1

TABLE 4. Computational time of cryptographic operations.

cryptographic operation	Description	Computational time	
		User	Server
T_h	One-way hash operation (SHA-512)	$\approx$ 1 ms	$\approx$ 0.01ms
T_{ch}	Chebyshev chaotic map operation	$\approx$ 1190 ms	$\approx$ 11.20 ms
T_s	Symmetric encryption operation (AES)	17.93 ms	0.16 ms
T_{pm}	Point multiplication/ modular multiplication operation	130 ms	1.17 ms
T_{pa}	Point addition operation	100 ms	0.1 ms
T_{minv}	Modular inverse operation	30 ms	0.3 ms
T_{me}	Modular exponentiation operation (RSA)	380 ms	3.16ms
T_f	Pseudo-random function operation	$\approx$ 1 ms	$\approx$ 0.01 ms

TABLE 5. Computational and communication cost in login and authentication phases.

Scheme	User computation	Server computation	Communication rounds
Khan et al. [20]	$3T_h \approx 3$ ms	$5T_h \approx 0.05$ ms	2
Chen et al. [21]	$5T_h \approx 5$ ms	$5T_h \approx 0.05$ ms	2
Jiang et al. [22]	$3T_h + T_s \approx 20.93$ ms	$3T_h + 3T_s \approx 0.19$ ms	2
Wu et al. [23]	$6T_h + 2T_s \approx 41.86$	$5T_h + 2T_s \approx 0.37$	3
Debiao et al. [24]	$5T_h + 1T_{me} \approx 385$ ms	$4T_h + 1T_{pa} + 1T_{minv} \approx 0.44$	3
Wei et al. [25]	$5T_h + 1T_{pm} + 1T_{me} \approx 515$ ms	$5T_h + 1T_{pm} + 1T_{me} \approx 3.27$ ms	3
Lee et al. [26]	$7T_h + 2T_{ch} \approx 2387$ ms	$8T_h + 2T_{ch} \approx 22.48$ ms	2
Farash et al. [28]	$5T_h + 4T_{pm} + 1T_{pa} \approx 625$ ms	$5T_h + 3T_{pm} \approx 3.54$ ms	3
Wen et al. [29]	$3T_h + 2T_s + 4T_{me} + 1T_{pm} \approx 1688.86$ ms	$2T_h + 2T_s + 4T_{me} + 1T_f \approx 12.99$ ms	3
Mishra et al. [37]	$5T_h + 1T_{ch} \approx 1195$ ms	$5T_h + 1T_{ch} \approx 11.25$ ms	2
kumari et al. [38]	$5T_h + T_s \approx 22.93$ ms	$3T_h + T_s \approx 0.19$ ms	2
Zhu et al. [39]	$4T_h + 1T_{me} \approx 384$ ms	$4T_h + 1T_{me} \approx 3.20$ ms	3
Das et al. [40]	$7T_h + 1T_{me} \approx 387$ ms	$7T_h + 1T_{me} \approx 3.23$ ms	3
Islam et al. [41]	$6T_h + 2T_{pm} \approx 266$ ms	$3T_h + T_{pm} \approx 1.20$ ms	2
Chaudhry et al. [42]	$5T_h + 3T_{pm} \approx 395$ ms	$3T_h + 1T_{pm} \approx 1.20$ ms	2
Jiang et al. [43]	$2T_h + T_s + 3T_{ch} \approx 3589.86$ ms	$1T_h + 2T_s + 3T_{ch} \approx 9.81$ ms	2
Tu et al. [44]	$5T_h + 4T_{pm} + 1T_{pa} \approx 625$ ms	$5T_h + 3T_{pm} \approx 3.56$ ms	3
Our scheme	$3T_h + 3T_s + 2T_{me} \approx 816.79$ ms	$3T_h + 3T_s + 2T_{me} \approx 6.83$ ms	2

```
Let U =
out (sch, ID);
in (sch, (AID2:bitstring,SCI2:bitstring,N2:bitstring,J2:bitstring,n2:
bitstriring,e2:bitstring)) ;
new r:bitsring;
let RPW=h((PW,r)) in
let L=xor(J2,h(RPW)) in
!
(
event UserStarted(ID);
new a:bitsring;
new T1:bitsring;
let RPW'=h((PW',r)) in
let J'=xor(L,h(RPW')) in
let F=mod(exp(J',a),n2) in
let C1=h(((T1,J',F))) in
let D=mod(exp(C1,e2),n2) in
let IZ=senc(((AID2,N2,SCI2, F)),C1) in

out(sch,(IZ,T1,D));
in(sch,(C2':bitstring,T2':bitstring,V':bitstring));
let W'=exp(V', a) in
let C2=senc(((T2, V', W')), C1) in
if C2'=C2'' then
let sk=h(T1, W', T2') in
0
).
```

FIGURE 7. The source code of the client agent.

and (AID_i, N, CS_i, F) respectively. It computes J', C_1', and checks the validity of C_1'. Then, it computes V, W, C_2, the session key SK, and sends the message (C_2, T_s, V) to U. The process S is modeled in Figure 6.

The user process defines the behavior of U, which computes RPW, J, F, C_1, D and IZ_i, and sends the message (IZ_i, T_i, D) via a public channel. When U receives a message (C_2, T_s, V), it computes C_2'' and its session key Sk. The process of U is modeled in Figure 7.

```
-- Query inj-event(UserAuthed(id)) ==> inj-event(UserStarted(id))
Completing...
Starting query inj-event(UserAuthed(id)) ==> inj-event(UserStarted(id))
RESULT inj-event(UserAuthed(id)) ==> inj-event(UserStarted(id)) is true.
-- Query not attacker(sk'[])
Completing...
Starting query not attacker(sk'[])
RESULT not attacker(sk'[]) is true.
-- Query not attacker(sk[])
Completing...
Starting query not attacker(sk[])
RESULT not attacker(sk[]) is true.
```

FIGURE 8. Execution results of the proposed two-factor mutual authentication protocol.

4) EXECUTION RESULTS

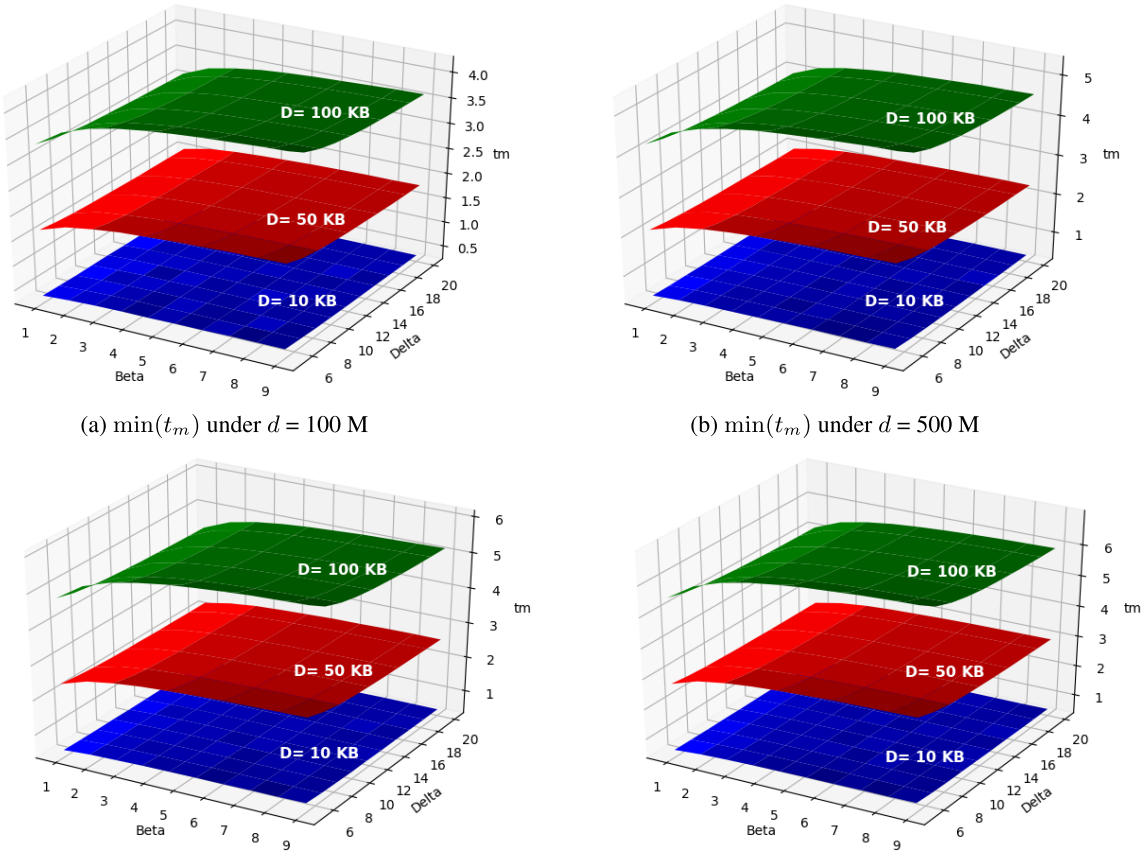
To verify the properties of the proposed two-factor mutual authentication scheme, we run the process presented in Figure 8 on ProVerif.

As shown in Figure 8, the results are interpreted as follows:

"RESULT not attacker (sk[]) is true": checks the secrecy property, i.e., the attacker failed to get Sk from the server.

"RESULT not attacker (sk'[]) is true": checks the secrecy property, i.e., the attacker failed to get the user's session key Sk'.

"RESULT inj-event (UserAuthed (id)) ⟹ inj-event (UserStarted (id)) is true": checks the authentication property, i.e., no intruder can get the user's identity during the scheme's execution. The user agent is therefore said to be authenticated by the server.



(a) $\min(t_m)$ under d = 100 M

(b) $\min(t_m)$ under d = 500 M

(c) $\min(t_m)$ under d = 1000 M

(d) $\min(t_m)$ under d = 2000 M

FIGURE 9. **min(t_m) under different distances.**

From the above results, we can deduce that the proposed two-factor mutual authentication scheme can ensure both secrecy and authentication properties.

VII. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed two-factor mutual authentication scheme and the offloading decision-making process.

A. TWO-FACTOR MUTUAL AUTHENTICATION SCHEME

We evaluate the efficiency of the proposed scheme, and compare it with other schemes [20]–[26], [28], [29], [37]–[44] with respect to computational and communication cost, which are incurred during the login and the authentication phases. As done in [53]–[56], we consider Philips HiPerSmart card 36MHz and Pentium IV 3GHZ to represent the user device and the server respectively. Table 4 shows the different cryptographic operations along with their computational time that are incurred by the user and the server. Although SHA-1 is used in [53]–[56], it is not secure as compared to SHA-256 and SHA-512. In [57], it has been reported that the execution time of SHA-256 is about 2.2 times (resp., 2 times) more than SHA-1 (resp., SHA-512), i.e., we consider that SHA-1 and SHA-512 both incur approximately the same computational time. As pseudorandom number generators can be built using hash functions, we consider that they incur approximately the same execution time. According to [58], the execution time of Chebyshev chaotic map operation $T_{ch} \approx 70\ T_s$. The XOR operation is omitted, due to its limited computational cost. To run the modular exponentiation operation, we choose RSA cryptographic technique, which is the standard public key cryptographic algorithm used in many communication protocols such as SSL/TLS, Secure Shell, etc., and does not depend on the public key infrastructure (PKI). Although the issue of PKI has been fixed by certificateless cryptography (CLC), it still requires a secured communication channel to transmit partial private keys to the users. In addition, RSA is still considered as a secure cryptographic algorithm, especially when using relatively large keys (2048-bit long keys). Finally, to execute the symmetric encryption/decryption operation, AES is selected as it is used in many protocols such as FTPS, HTTPS, SFTP, etc.

From the comparison results, which are outlined in Table 5, we can notice that our authentication scheme requires a computational cost of $3T_h + 3\ T_s + 2\ T_{me}$ and $3\ T_h + 3\ T_s + 2\ T_{me}$ on the user side and the server side, respectively. It also needs 2 messages to execute the login and the authentication phases. Compared to Jiang et al. scheme [22], our scheme incurs an additional cost of 2 symmetric encryption/decryption

TABLE 6. Simulation parameters.

Parameter	Value
$TR_{m,B}$	56.37 ($d = 100m$), 44.44 ($d = 500m$), 39.3 ($d = 1000m$), 34.16 ($d = 2000m$)
d	100m, 500m, 1000m, 2000m
β	S_m^{Auth} to $10 \times S_m^{Auth}$
S_m^{Auth}	0.9 watt
$P_{m,T}$	0.545 watt
$P_{m,R}$	0.090 watt
D	10KB, 50KB, 100KB
Δ	5 to 20

operations and 2 modular exponentiation operations (resp., 2 modular exponentiation operations) on the user side (resp., server side). On the other hand, our scheme incurs less computational cost compared to the schemes proposed by Lee et al. [26], Wen et al. [29], Mishra et al. [37], and Jian et al. [43]. Although our scheme is not the best in terms of computational efficiency, it provides better security, as shown in Table 3.

B. OFFLOADING DECISION-MAKING PROCESS

We evaluate the offloading decision-making process by computing the lower-bound of t_m (i.e., $\min(t_m)$) required to perform the offloading. To do so, we vary different parameters: Δ, D, d, R, S_m^{Auth}, and β and generate different plots, as presented in Figure 9. The simulation parameters used are presented in Table 6.

The lower-bound of t_m are shown as a function of Δ and β and under different values of distance d. From the results presented in Figure 9a, where we choose $d = 100$ meters, the obtained values of t_m are 0.42s, 2.11s, and 4.22s under D = 10KB, 50KB, and 100 KB respectively. In Figure 9d, d is set to 2000 meters. The obtained values of t_m are 0.7s, 3.48s, and 6.96s under $D = 10$KB, 50KB, and 100 KB respectively. From these results, we can see that the highest variation occurs when the amount of transferred data is increased/decreased. However, the variation of the distance between the mobile device and the mobile base station has a lower impact on t_m. Variation of β (i.e., energy dissipation of apps running on the mobile device) also has a low impact on t_m, as the CPU consumes very less energy compared to data communication. On the other hand, by increasing Δ, t_m decreases, which means that the decision-making process is more towards performing offloading.

VIII. CONCLUSION

In this paper, we have proposed a novel mobile cloud offloading architecture for the two-factor mutual authentication application. The architecture leverages the security features that are offered by Android OS to propose a two-factor mutual authentication scheme based on a novel mechanism, named virtual smart card. The architecture also comprises a decision-making process to offload the authentication application and its virtual smart card, based on three conditions: security, mobile device's residual energy, and energy cost. From the energy cost equation, which considers different heterogeneous parameters, the offloading decision-making factor $\min(t_m)$ is derived. The proposed two-factor mutual authentication scheme has been analyzed in terms of efficiency, as well as against different attacks. It has also been formally verified using BAN Logic and ProVerif techniques. In addition, the decision-making process has been evaluated by computing the different values of $\min(t_m)$ under various parameters. The results have shown that the size of the offloaded data highly affects the offloading decision.

REFERENCES

[1] *Smartphone Market Share*. Accessed: Jan. 20, 2020. [Online]. Available: https://www.idc.com/promo/smartphone-market-share/os

[2] S.-H. Hung, C.-S. Shih, J.-P. Shieh, C.-P. Lee, and Y.-H. Huang, "Executing mobile applications on the cloud: Framework and issues," *Comput. Math. Appl.*, vol. 63, no. 2, pp. 573–587, Jan. 2012.

[3] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Netw. Appl.*, vol. 18, no. 1, pp. 129–140, Feb. 2013.

[4] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[5] M. B. Mollah, M. A. K. Azad, and A. Vasilakos, "Security and privacy challenges in mobile cloud computing: Survey and way ahead," *J. Netw. Comput. Appl.*, vol. 84, pp. 38–54, Apr. 2017.

[6] K. Akherfi, M. Gerndt, and H. Harroud, "Mobile cloud computing for computation offloading: Issues and challenges," *Appl. Comput. Inform.*, vol. 14, no. 1, pp. 1–16, Jan. 2018.

[7] A. Lioy, A. Pastor, F. Risso, R. Sassu, and A. L. Shaw, "Offloading security applications into the network," in *Proc. Conf. eChallenges*, 2014, pp. 1–9.

[8] Y. Li, J. Liu, Q. Li, and L. Xiao, "Mobile cloud offloading for malware detections with learning," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2015, pp. 197–201.

[9] R. Bonafiglia, F. Ciaccia, A. Lioy, M. Nemirovsky, F. Risso, and T. Su, "Offloading personal security applications to a secure and trusted network node," in *Proc. 1st IEEE Conf. Netw. Softwarization (NetSoft)*, Apr. 2015, pp. 1–2.

[10] S. A. Saab, F. Saab, A. Kayssi, A. Chehab, and I. H. Elhajj, "Partial mobile application offloading to the cloud for energy-efficiency with security measures," *Sustain. Comput., Inform. Syst.*, vol. 8, pp. 38–46, Dec. 2015.

[11] Y. Xia, Y. Liu, C. Tan, M. Ma, H. Guan, B. Zang, and H. Chen, "*TinMan*: Eliminating confidential mobile data exposure with security oriented offloading," in *Proc. 10th Eur. Conf. Comput. Syst.*, 2015, p. 27.

[12] (2016). *Android Malware Defeats Two-Factor Authentication*. Accessed: Jan. 20, 2020. [Online]. Available: https://www.welivesecurity.com/2016/03/09/android-trojan-targets-online-banking-users/

[13] (2016). *Android Banking Trojan Masquerades as Flash Player and Bypasses 2FA*. Accessed: Jan. 20, 2020. [Online]. Available: https://www.welivesecurity.com/2016/03/09/android-trojan-targets-online-banking-users/

[14] M. U. Aslam, A. Derhab, K. Saleem, H. Abbas, M. Orgun, W. Iqbal, and B. Aslam, "A survey of authentication schemes in telecare medicine information systems," *J. Med. Syst.*, vol. 41, no. 1, p. 14, 2017.

[15] M. A. Ferrag, L. Maglaras, and A. Derhab, "Authentication and authorization for mobile IoT devices using biofeatures: Recent advances and future trends," *Secur. Commun. Netw.*, vol. 2019, pp. 1–20, May 2019.

[16] M. A. Ferrag, L. Maglaras, A. Derhab, and H. Janicke, "Authentication schemes for smart mobile devices: Threat models, countermeasures, and open research issues," *Telecommun. Syst.*, vol. 73, no. 2, pp. 317–348, Feb. 2020.

[17] C. Chang and T. Wu, "A password authentication scheme without verification tables," in *Proc. 8th IASTED Int. Symp. Appl. Inform.*, Innsbruck, Austria, 1990, pp. 202–204.

[18] M. L. Das, A. Saxena, and V. P. Gulati, "A dynamic id-based remote user authentication scheme," *IEEE Trans. Consum. Electron.*, vol. 50, no. 2, pp. 629–631, Jul. 2004.

[19] Y.-Y. Wang, J.-Y. Liu, F.-X. Xiao, and J. Dan, "A more efficient and secure dynamic ID-based remote user authentication scheme," *Comput. Commun.*, vol. 32, no. 4, pp. 583–585, Mar. 2009.

[20] M. K. Khan, S.-K. Kim, and K. Alghathbar, "Cryptanalysis and security enhancement of a 'more efficient & secure dynamic ID-based remote user authentication scheme,'" *Comput. Commun.*, vol. 34, no. 3, pp. 305–309, 2011.
[21] H.-M. Chen, J.-W. Lo, and C.-K. Yeh, "An efficient and secure dynamic ID-based authentication scheme for telecare medical information systems," *J. Med. Syst.*, vol. 36, no. 6, pp. 3907–3915, Dec. 2012.
[22] Q. Jiang, J. Ma, Z. Ma, and G. Li, "A privacy enhanced authentication scheme for telecare medical information systems," *J. Med. Syst.*, vol. 37, no. 1, pp. 1–8, 2013.
[23] Z.-Y. Wu, Y.-C. Lee, F. Lai, H.-C. Lee, and Y. Chung, "A secure authentication scheme for telecare medicine information systems," *J. Med. Syst.*, vol. 36, no. 3, pp. 1529–1535, Jun. 2012.
[24] H. Debiao, C. Jianhua, and Z. Rui, "A more secure authentication scheme for telecare medicine information systems," *J. Med. Syst.*, vol. 36, no. 3, pp. 1989–1995, Jun. 2012.
[25] J. Wei, X. Hu, and W. Liu, "An improved authentication scheme for telecare medicine information systems," *J. Med. Syst.*, vol. 36, no. 6, pp. 3597–3604, Dec. 2012.
[26] T.-F. Lee, "An efficient chaotic maps-based authentication and key agreement scheme using smartcards for telecare medicine information systems," *J. Med. Syst.*, vol. 37, no. 6, pp. 1–9, 2013.
[27] L. Zhang, S. Tang, and Z. Cai, "Efficient and flexible password authenticated key agreement for voice over Internet protocol session initiation protocol using smart card," *Int. J. Commun. Syst.*, vol. 27, no. 11, pp. 2691–2702, 2014.
[28] M. S. Farash, "Security analysis and enhancements of an improved authentication for session initiation protocol with provable security," *Peer-Peer Netw. Appl.*, vol. 9, no. 1, pp. 82–91, Jan. 2016.
[29] F. Wen and D. Guo, "An improved anonymous authentication scheme for telecare medical information systems," *J. Med. Syst.*, vol. 38, no. 5, pp. 1–11, 2014.
[30] H.-T. Lee, D. Kim, M. Park, and S.-J. Cho, "Protecting data on Android platform against privilege escalation attack," *Int. J. Comput. Math.*, vol. 93, no. 2, pp. 401–414, Feb. 2016.
[31] F. Boukayoua, J. Lapon, B. De Decker, and V. Naessens, "Secure storage on Android with context-aware access control," in *Proc. IFIP Int. Conf. Commun. Multimedia Secur.* Berlin, Germany: Springer, 2014, pp. 46–59.
[32] T. Cooijmans, J. de Ruiter, and E. Poll, "Analysis of secure key storage solutions on Android," in *Proc. 4th ACM Workshop Secur. Privacy Smartphones Mobile Devices*, 2014, pp. 11–20.
[33] M. Allalouf, R. Ben-Av, and A. Gerdov, "StoreDroid: Sensor-based data protection framework for Android," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Aug. 2014, pp. 511–517.
[34] X. Li, H. Hu, G. Bai, Y. Jia, Z. Liang, and P. Saxena, "DroidVault: A trusted data vault for Android devices," in *Proc. 19th Int. Conf. Eng. Complex Comput. Syst.*, 2014, pp. 29–38.
[35] M. Sun, M. Li, and J. Lui, "DroidEagle: Seamless detection of visually similar Android apps," in *Proc. 8th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, 2015, p. 9.
[36] C. V. Anamuro, N. Varsier, J. Schwoerer, and X. Lagrange, "Simple modeling of energy consumption for D2D relay mechanism," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2018, pp. 231–236.
[37] D. Mishra, J. Srinivas, and S. Mukhopadhyay, "A secure and efficient chaotic map-based authenticated key agreement scheme for telecare medicine information systems," *J. Med. Syst.*, vol. 38, no. 10, p. 120, Oct. 2014.
[38] S. Kumari, M. K. Khan, and R. Kumar, "Cryptanalysis and improvement of 'a privacy enhanced scheme for telecare medical information systems,'" *J. Med. Syst.*, vol. 37, no. 4, p. 9952, 2013.
[39] Z. Zhu, "An efficient authentication scheme for telecare medicine information systems," *J. Med. Syst.*, vol. 36, no. 6, pp. 3833–3838, Dec. 2012.
[40] A. K. Das and B. Bruhadeshwar, "An improved and effective secure password-based authentication and key agreement scheme using smart cards for the telecare medicine information system," *J. Med. Syst.*, vol. 37, no. 5, pp. 1–17, 2013.
[41] S. H. Islam and M. K. Khan, "Cryptanalysis and improvement of authentication and key agreement protocols for telecare medicine information systems," *J. Med. Syst.*, vol. 38, no. 10, pp. 1–16, 2014.
[42] S. A. Chaudhry, H. Naqvi, T. Shon, M. Sher, and M. S. Farash, "An improved and secure biometric authentication scheme for telecare medicine information systems based on elliptic curve cryptography," *J. Med. Syst.*, vol. 39, no. 6, pp. 1–11, 2015.
[43] Q. Jiang, J. Ma, X. Lu, and Y. Tian, "Robust chaotic map-based authentication and key agreement scheme with strong anonymity for telecare medicine information systems," *J. Med. Syst.*, vol. 38, no. 2, pp. 1–8, 2014.
[44] H. Tu, N. Kumar, N. Chilamkurti, and S. Rho, "An improved authentication protocol for session initiation protocol using smart card," *Peer-Peer Netw. Appl.*, vol. 8, no. 5, pp. 903–910, Sep. 2015.
[45] F. Wen, "A more secure anonymous user authentication scheme for the integrated EPR information system," *J. Med. Syst.*, vol. 38, no. 5, pp. 1–7, 2014.
[46] M. Masdari and S. Ahmadzadeh, "A survey and taxonomy of the authentication schemes in telecare medicine information systems," *J. Netw. Comput. Appl.*, vol. 87, pp. 1–19, Jun. 2017.
[47] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," *Proc. Roy. Soc. London A, Math. Phys. Sci.*, vol. 426, no. 1871, pp. 233–271, 1989.
[48] S. Kodra, "Protocol verification with ProVerif," Univ. Tartu, Tartu, Estonia, Tech. Rep., 2015.
[49] B. Blanchet, "Automatic verification of correspondences for security protocols," *J. Comput. Secur.*, vol. 17, no. 4, pp. 363–434, 2009.
[50] C. J. Cremers, "The Scyther Tool: Verification, falsification, and analysis of security protocols," in *Proc. Int. Conf. Comput. Aided Verification*. Berlin, Germany: Springer, 2008, pp. 414–418.
[51] A. Armando, W. Arsac, T. Avanesov, M. Barletta, A. Calvi, A. Cappai, R. Carbone, Y. Chevalier, L. Compagna, and J. Cuéllar, "The AVANTSSAR platform for the automated validation of trust and security of service-oriented architectures," in *Proc. Int. Conf. Tools Algorithms Construction Anal. Syst.* Berlin, Germany: Springer, 2012, pp. 267–282.
[52] A. Trieu, A. R. A. Chatalic, B. Tessiau, and L. S. S. Kachanovich, "Aide à la conception et vérification de spécifications formelles de protocoles cryptographiques," Univ. Rennes, Rennes, France, Tech. Rep., 2014, vol. 1.
[53] R. Amin, S. H. Islam, G. Biswas, M. K. Khan, and N. Kumar, "An efficient and practical smart card based anonymity preserving user authentication scheme for TMIS using elliptic curve cryptography," *J. Med. Syst.*, vol. 39, no. 11, p. 180, 2015.
[54] L. Xu and F. Wu, "An improved and provable remote user authentication scheme based on elliptic curve cryptosystem with user anonymity," *Secur. Commun. Netw.*, vol. 8, no. 2, pp. 245–260, Jan. 2015.
[55] S. Roy, S. Chatterjee, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "On the design of provably secure lightweight remote user authentication scheme for mobile cloud computing services," *IEEE Access*, vol. 5, pp. 25808–25825, 2017.
[56] S. Qiu, G. Xu, H. Ahmad, and L. Wang, "A robust mutual authentication scheme based on elliptic curve cryptography for telecare medical information systems," *IEEE Access*, vol. 6, pp. 7452–7463, 2018.
[57] S. Gueron, S. Johnson, and J. Walker, "SHA-512/256," in *Proc. 8th Int. Conf. Inf. Technol., New Generat.*, Apr. 2011, pp. 354–358.
[58] X. Wang, S. Wang, Z. Wang, and M. Zhang, "A new key agreement protocol based on Chebyshev chaotic maps," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5028–5035, Dec. 2016.

ABDELOUAHID DERHAB received the Engineer's, M.Sc., and Ph.D. degrees in computer science from the University of Sciences and Technology Houari Boummediene (USTHB), Algiers, in 2001, 2003, and 2007, respectively. He was a Computer Science Engineer and a full-time Researcher with the CERIST Research Center, Algeria, from 2002 to 2012. He was an Assistant Professor with King Saud University, from 2012 to 2018. He is currently an Associate Professor with the Center of Excellence in Information Assurance (COEIA), King Saud University. He also served as a workshop chair, a technical committee chair, and a reviewer for many journals and international conferences. He is the author of more than 100 articles in different peer-reviewed journals and conferences. He is also a cyber-security policy analyst at Global Foundation for Cyber Studies and Research (GFCYBER). His research interests are malware analysis, network security, intrusion detection, mobile security, the Internet of Things, smart grid, blockchain, and cyber security policies. He served as a Lead Guest Editor and on the editorial board for some peer-reviewed journals.

MOHAMED BELAOUED received the master's and Ph.D. degrees in computer science from the University of Skikda, in 2011 and 2016, respectively. He is currently an Associate Professor with the University of Constantine 1, Algeria, and a Researcher with the LIRE Laboratory, Software Technologies and Information Systems Department, University of Constantine 2. His research interests include malware analysis and detection, intrusion detection, and networks and the IoT security.

MOHAMED GUERROUMI received the Ph.D. degree from USTHB University, in 2014. He was a Data Network Senior Engineer with the Department of Core Network, Orascom Telecom Algerie, Algiers, from 2006 to 2013. From January 2014 to October 2016, he was a consultant as an IP Network and Security Designer Expert with DIVONA Company. He is currently an Associate Professor with the Department of Computer science, USTHB University, Algiers, Algeria. His research interests are in performance evaluation of multihop wireless networks, focusing on developing performance evaluation frameworks and designing protocols in MAC and routing layers for wireless sensor networks and vehicular ad hoc networks (VANET). Also, his current research interest includes the security policies in wireless sensor networks and mobile and SDN networks. Dr. Guerroumi served in the TPC for IEEE GLOBCOM, IEEE ICC, ANT, ANT2, Dependsys, ISCC, SpaCC, a Program Chair of DependSys, and a Publication Chair of SGIo. Also, he served as a reviewer in numerous international journals, such as *Future Generation Computer Systems* (Elsevier), the IEEE Communications Magazine, *Computers & Electrical Engineering* (Elsevier) and *Wireless Personal Communications* (Springer). He served as a Guest Editor for *Annals of Telecommunications* Journal and a workshop chair for several workshops.

FARRUKH ASLAM KHAN (Senior Member, IEEE) received the M.S. degree in computer system engineering from the GIK Institute of Engineering Sciences and Technology, Pakistan, in 2003, and the Ph.D. degree in computer engineering from Jeju National University, South Korea, in 2007. He is currently a Professor with the Center of Excellence in Information Assurance, King Saud University, Riyadh, Saudi Arabia. He is also a Founding Director of the Wireless Networking and Security Research Group, National University of Computer and Emerging Sciences, Islamabad, Pakistan. He has over 100 publications in refereed international journals and at conferences. He has co-organized several international conferences and workshops. He has successfully supervised four Ph.D. students and sixteen M.S. thesis students. Several M.S. and Ph.D. students are also working under his supervision. His research interests include cybersecurity, body sensor networks and e-health, bio-inspired and evolutionary computation, and the Internet of Things. He is on the panel of reviewers of over 30 reputed international journals and numerous international conferences. He also received professional training from the Massachusetts Institute of Technology, New York University, IBM, and other professional institutions. He is a Fellow of the British Computer Society (BCS). He serves as an Associate Editor for prestigious international journals, including the IEEE Access, *PLOS One*, *Neurocomputing* (Elsevier), *Ad Hoc and Sensor Wireless Networks*, *KSII Transactions on Internet and Information Systems*, *Human-Centric Computing and Information Sciences* (Springer), and *Complex & Intelligent Systems* (Springer).