- ORIGINAL ARTICLE -

A Client-based User Authentication Scheme for the Cloud of Things Environment

Un esquema de Autenticación de Usuario basado en el Cliente Para el Entorno de la Nube de las Cosas

Norliza Katuk[1] Roberto Vergallo[2] Tito Sugiharto[3] and Rio Andriyat Krisdiawan[3]
[1]*Universiti Utara Malaysia, Malaysia*
k.norliza@uum.edu.my
[2]*University of Salento, Italy*
roberto.vergallo@unisalento.it
[3]*Universitas Kuningan, Indonesia*
{tito, rioandriyat}@uniku.ac.id

Abstract

The limited capabilities of IoT devices have resulted in some of the tasks of IoT applications being distributed to a cloud server, which witnessed the arisen of the cloud of things (COT). It enables IoT applications' development and deployment as a service, providing additional data storage, enhanced processing performance, and fast communication between devices. As COT involves communication between IoT devices, a remote server, and users, remote user authentication is crucial to meeting security demands. Therefore, this study designs a client-based user authentication scheme utilizing smartphone fingerprint recognition technology to fill the gap. The scheme comprises six phases, namely (i) configuration phase, (ii) enrolment phase, (iii) authentication phase, (iv) password update phase, (v) fingerprint revocation phase, and (vi) smartphone revocation phase. The security analysis and automated verification using ProVerif suggested that the scheme is resistant to user impersonating attacks, replay attacks, and man-in-the-middle attacks. The study's outcome could help secure user credentials from attacks on applications that involve IoT and the cloud.

Keywords: cryptography, internet of things, sensors, authentication, encryption

Resumen

Las capacidades limitadas de los dispositivos IoT han dado como resultado que algunas de las tareas de las aplicaciones IoT se distribuyan a un servidor en la nube, lo que es testigo del surgimiento de la Nube de las Cosas (COT). Esta permite el desarrollo y la implementación de aplicaciones IoT como un servicio, proporcionando almacenamiento de datos adicional, mayor rendimiento de procesamiento y comunicación rápida entre dispositivos. Dado que la COT implica la comunicación entre dispositivos IoT, un servidor remoto y usuarios, la autenticación de usuarios remotos es crucial para satisfacer las demandas de seguridad. Por lo tanto, este estudio diseña un esquema de autenticación de usuario basado en el cliente que utiliza tecnología de reconocimiento de huellas digitales en teléfonos inteligentes para colmar la brecha. El esquema consta de seis fases: (i) fase de configuración, (ii) fase de inscripción, (iii) fase de autenticación, (iv) fase de actualización de contraseña, (v) fase de revocación de huellas digitales y (vi) fase de revocación de teléfonos inteligentes. A partir del análisis de seguridad y la verificación automatizada con ProVerif surge que el esquema es resistente a diferentes ataques, por ejemplo ataques de suplantación de identidad del usuario, los ataques de repetición y los ataques man-in-the-middle. El resultado del estudio podría ayudar a proteger las credenciales de los usuarios de los ataques a las aplicaciones que involucran IoT y la nube.

Palabras claves: Criptografía, Internet de las cosas, Sensores, Autenticación, Encriptación

1. Introduction

Internet of Things (IoT) is a network of various sensing devices that provides services according to application integration [1-3]. It has been employed for data capturing devices in various domains, including transportation, infrastructure, computing intelligence, and e-health, just to name a few [4]. The sensing devices generate massive data that can be analyzed to form intelligent environments like smart cities [5], smart homes [6, 7], and smart vehicles [8]. IoT has also merged with cloud computing which forms a domain known as the cloud of things (COT) [9-14]. In COT, the cloud server acts as an IoT backend system that receives data from the sensors,

processes them, and stores them centrally [13, 14].

COT arises for monitoring and controlling IoT devices within mobile cloud computing to support advanced applications [9, 15]. IoT devices' limited capabilities have resulted in some IoT applications being distributed to a cloud server [16]. Integration of IoT with cloud computing has become a necessary technology to manage such data for creating more valuable and intelligent services and applications [9]. COT also enables the development and deployment of IoT applications as a service, additional data storage, enhanced processing performance, and fast communication between devices [11]. COT's emergence can create beneficial services for humans; however, these services face significant security threats [17]. Further, limited research has been done on the shortcomings of IoT and cloud integration, especially in data security [11].

Researchers believe that communication security remains a significant problem in the COT environment. The reason is that the information remains prone to attacks conducted through SQL injections, eavesdropping, man-in-the-middle, and many other methods [11]. As COT involves communication between a remote server and IoT devices to establish connectivity in any physical environment, remote user authentication is crucial to meet security demands [4]. Authentication ensures the message is received from a legitimate sender and serves as a front-line defence of the IoT network from unauthorized access. Thus, authentication is considered an essential requirement for IoT [18, 19]. Attacks on confidentiality and authentication, as well as the availability of network services, are often significant issues in the security of IoT networks [20]. In addition, COT requires storage, processing, and energy capacity for security protection. Unfortunately, the current security scheme that uses traditional encryption requires a sizeable computational resource [17]. Conventional encryption is not suitable for IoT environments due to limited resources, resulting in lightweight authentication schemes suitable for IoT environments [18].

Thanks to mobile computing, they can support IoT devices' security, primarily through hardware such as smartphones [21]. Smartphones can be used as a user-friendly authentication tool; built based on the existing ecosystem that defines and enforces custom security policies necessary for IoT devices [21]. Nowadays, the smartphone plays beyond a communication device [22], especially with many embedded sensors that enable applications in various domains, including IoT. It appears necessary for everyone at all levels of age [22]. Smartphones act as communication, entertainment, socializing, shopping, educational, and personal organizing tools. Smartphones also catalyzed access to cloud-based mobile applications by millions of users [23]. As IoT contributes to the Internet ecosystem, applications or services accessible via smartphones benefit users [14]. Smartphones have been used to interact with IoT devices to access data and control them [6]. Further, a smartphone's built-in camera and fingerprint scanner could be used as user biometric authentication for IoT-based applications.

The robust nature of biometric authentication has led to its significant deployment in diverse domains [24]. Biometric security systems provide secure access compared to alphanumeric-based passwords [25]. It can also be strengthened using multi-factor authentication [26]. Many authentication schemes based on biometric systems have been proposed, like hand-based multibiometric [27], fingerprint [22], and face image [24]. Nonetheless, the fingerprint is a reliable method of user authentication as it is unique to each individual, thus making it efficient to authenticate users [28]. It is also a suitable authentication method for accessing COT-based applications through smartphones. This study aims to design an authentication scheme for accessing COT applications using smartphones as the client interface to address the authentication scheme's limitation for COT-based applications. The following sections of this article describe the proposed scheme's design and evaluation.

2. Related Work

Impersonating an authorized user is one of the most critical threats to any computer system. This threat can be addressed using proper user authentication [29] by verifying the communicating parties' true identities [30]. In the context of computer-based systems, authentication is verifying the identity of a user, device, or other entity requesting access to a computer system [29]. Authentication methods can be classified into three, namely (1) methods based on human memory such as passwords and personal identification numbers, (2) methods based on physical devices such as USB cards, magnetic or IC cards, and (3) methods based on biometrics such as fingerprint, and iris [31-33]. The authentication method by itself sends users' authentication information to the server in the form of plaintext. Hackers can easily view confidential information by eavesdropping on the communication line during the authentication phase. It could lead to a masquerading attack (i.e., impersonating legitimate users). This kind of attack is difficult to identify unless abnormal activities are detected from the stolen user profiles. Hence, the authentication methods require additional mechanisms, specifically encryption methods, to conceal the original form of users' authentication information into an encrypted form. A combination

of the authentication methods and the encryption mechanism presented systematically and clearly, is referred to as an authentication scheme.

SAP [34], a European multinational software corporation, describes an authentication scheme as "a definition of what is required for an authentication process". In a more specific definition, an authentication scheme specifies the authentication method, the protocol, the process for authenticating users, and possible algorithms for verifying the users' identity to access resources from a computer system [35]. The authentication scheme is one of the critical security mechanisms and a crucial requirement for computer systems. It provides user authentication and protects user authentication information from being leaked [36], consequently assisting in user experience optimization [16]. Studies on authentication schemes have long started. As a result, many improvements have been made to cater to the needs of various types of computer systems. For example, the password is the most widely used and straightforward user authentication method in a distributed computer system environment. However, the method is vulnerable to password-guessing attacks since many users create an easy-to-remember password [37]. Therefore, an authentication method could be insufficient to provide a secure user authentication process. Therefore researchers designed an improved authentication scheme that combined the method with encryption, such as Liao et al. [38], that proposed a password authentication scheme by using Diffie–Hellman key agreement protocol to strengthen the security of passwords.

Authentication schemes that are designed using single encrypted authentication methods are called single-factor authentication schemes [39]. On the other hand, a combination of two or more schemes is known as multi-factor authentication schemes [40-42]. Ometov et al. [33] provided a comprehensive review of authentication schemes' evolution from single-factor to multiple-factor. Threats to authentication information are rising exponentially, and recent security incidents have demonstrated that a single-factor authentication scheme is insufficient [42]. Multi-factor authentication schemes are more secure than the single-factor authentication scheme in a distributed computer systems environment [41]. There is also a need for multi-factor authentication schemes that provide users with different authentication choices [43].

Multi-factor authentication schemes are designed to combat security attacks in the corresponding domains, carefully considering the hardware, software, and network features that form the computer systems. For example, multi-factor authentication schemes designed for wireless sensor networks and IoT should be secure and lightweight to cope with sensors' features limited in power and processing capabilities [35]. Based on this example, it can be said that a multi-factor authentication scheme that works on a particular domain of a computer system does not necessarily perform similarly in other environments. Therefore, the design for multi-factor authentication schemes should tailor to the intended computer systems or applications' requirements.

Authentication schemes generally have specific requirements, including mutual authentication, confidentiality, anonymity, availability, forward secrecy, scalability, and attack resistance [35]. However, most importantly, authentication schemes for IoT should be lightweight because IoT networks are resource-constrained and are limited in processing power, battery backup, memory, and speed [35]. As IoT has evolved and received attention among researchers, most existing authentication schemes are designed for IoT devices, not for user authentication for IoT-based systems, especially COT. For example, Kalra and Sood [44] proposed an authentication scheme using HyperText Transfer Protocol cookies and Public Key Cryptography using Elliptic Curve Cryptography algorithms for embedded devices and cloud servers. However, the scheme does not provide mutual authentication and lacks a session-key agreement [6]. Due to this limitation, Chang et al. [6] improved the scheme by simplifying the whole process from five phases into three phases, namely (1) the registration phase, (2) the pre-computation and login phase, and (3) the authentication phase. Other than this, Gope et al. [45] employed a radio frequency identification (RFID) tag with a hash function for encryption. The scheme has the potential in terms of its performance; however, using RFID tags for authenticating IoT devices is exposed to cloning attacks.

IoT is unlikely to fade anytime soon, and designing lightweight cryptographic schemes suitable for user authentication in IoT remains a research challenge [46]. Further, many applications are now being developed so that IoT devices can connect to private cloud servers for data storage [47]. Connecting IoT devices to the cloud server created a new computer system environment named COT, requiring a different authentication scheme to suit this setting's requirements. Literature analysis prepared for this research revealed that a multi-factor authentication scheme for the COT environment is minimal. Amin et al. [47] proposed passwords and smart cards with hash and XOR as their authentication mechanism. Performance analysis on the proposed scheme outperformed selected single-factor authentication schemes in terms of computation, storage, and communication cost.

However, studies have also proven that smart cards may expose users to offline password guessing attacks and smart-card forgery attacks

with the lost/stolen smart cards [48]. The major limitation in the existing multi-factor user authentication scheme could expose users to a severe security threat within the COT environment, including masquerading or impersonation attack that penetrates systems using the lost/stolen smart cards. Yu et al. [49] emphasized that authentication and key agreement protocols are the most crucial aspect of COT security. They proposed an improved authentication and key agreement scheme based on He et al.'s [50] scheme. Fig. 1 illustrates the user authentication model within the IoT-cloud environment.

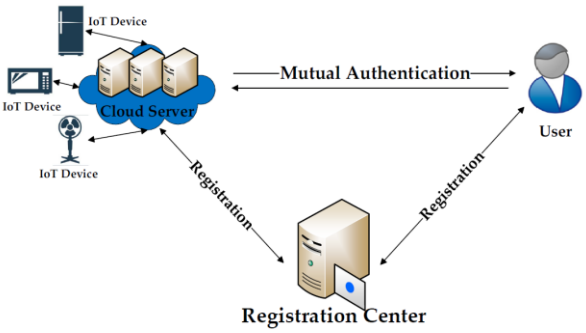

Fig. 1 Authentication in an IoT-based cloud environment [49].

3. The Proposed User Authentication Scheme

3.1 Security Requirements

The proposed authentication scheme intends to protect the user credential data and communication security of the COT system. Therefore, the proposed authentication scheme should fulfil the following critical security requirements:

1. Confidentiality: The user credential data stored in the cloud server should be protected against any disclosure or unauthorized access.
2. Integrity: The user credential data stored in the cloud server should not be modified and not compromised either during the communication process or storage.
3. Availability: The user credential data stored at the cloud server and server should always run correctly and be available at any time the authorized users need. This requirement encounters denial-of-service attacks.
4. Anonymity: The user and the cloud server identities should be concealed during the communication so that a malicious person cannot trace the identity of the communication parties. When the identities are not known, it avoids identity stolen. Therefore, this requirement can protect the COT system against user impersonating attacks.
5. Mutual authentication: The user and the cloud server should be able to authenticate the identity of each other to avoid impersonating attacks either as a legitimate user or a legitimate cloud server.
6. Data freshness: Communicated user credential data between the cloud server and the user should be fresh. Resending data between the communicating parties should be avoided to protect the systems from replay attacks.
7. Forward secrecy: The malicious person cannot decrypt a message using the previously transmitted data from the user and the cloud server during their communication.

3.2 Threat Models

A COT system may be exposed to the following threats:

1. Denial-of-service attacks: Malicious persons create traffic to the cloud server that causes flood and service breakdown.
2. User impersonating attacks: Malicious persons use legitimate users' credentials to gain access to COT systems.
3. Replay attacks: Malicious persons resend the communicated message either to the user or the cloud server.
4. Man-in-the-middle attacks: Malicious persons capture the message, manipulate, and resend the message like a legitimate user.

3.3 Notations, Assumptions, and System Model

The notations in Table 1 are used to describe the proposed user authentication scheme and protocol for the COT systems.

Table 1. The nations used for the proposed user authentication scheme.

Symbol	Description
U	*The user*
Id_U	*The unique identity of U*
PW	*The password of U*
E_{PW}	*Encrypted PW*
D_{PW}	*Decrypted PW*
F	*The fingerprint template of U*
E_F	*Encrypted F*
D_F	*Decrypted F*
SP	*The smartphone for U fingerprint reading*
Id_{SP}	*The unique identity of SP*
C	*The connection module*
CS	*The cloud server*
Id_{CS}	*The unique identity of CS*
I	*The IoT device*
Id_I	*The unique identity of I*
G	*The gateway connecting I and CS*
Id_G	*The unique identity of G*
A	*The client IoT-Cloud module on SP*
MS	*An encrypted communication message*
MS_{Id}	*The unique identification of MS*
MK	*A secret key of G*
SK	*A shared secret key of CS and G*
CK	*A shared secret key between SP and CS*

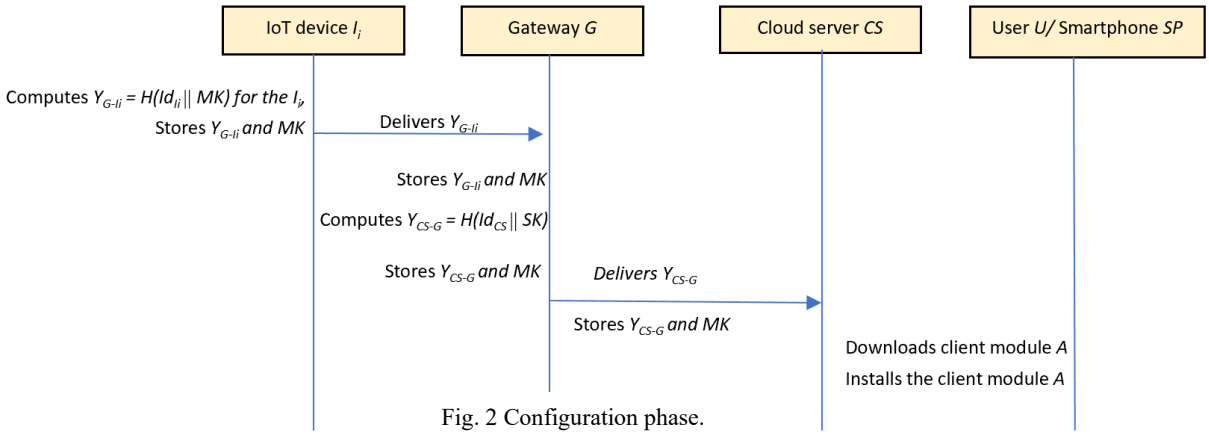

Fig. 2 Configuration phase.

L	*An activation link*
AL	*Authenticated L*
T	*Timestamp*
$h(.)$	*A one-way hash function*
H	*The value derived from h(.)*
$\oplus$	*A bitwise XOR operation*
$\|\|$	*Concatenation*

The proposed authentication scheme has the following assumptions:

1. The scheme covers user credential data for the authentication in accessing the COT system. However, this scheme does not cover communication between the IoT devices with the cloud server.
2. The shared secret key is sent through a secure link.
3. A secure channel is used between the smartphone and cloud server communication.

3.4 Authentication Protocol

The proposed authentication scheme for COT systems contains six phases, namely (i) Configuration phase, (ii) Enrolment phase, (iii) Authentication phase, (iv) Password update phase, (v) Fingerprint revocation phase, and (vi) Smartphone revocation phase.

Configuration Phase

The initial phase ensures a secure and trusted establishment of a connection between IoT devices and a cloud server. Next, a gateway is needed to connect both parties to form a COT system. Fig. 2 illustrates the configuration phase.

Step C1: The IoT device I_i is connected to gateway G by the connection module C. The connection module computes $Y_{G\text{-}Ii} = H(Id_{Ii} || MK)$ for the Ii, for $1 \leq i \leq n$, where n is the number of IoT devices. The IoT device and the gateway store both values of $Y_{G\text{-}Ii}$ and MK.

Step C2: The connection module connects gateway G to the cloud server CS. The connection module computes $Y_{CS\text{-}G} = H(Id_{CS} || SK)$ for the cloud server. The gateway and the cloud server store both values of $Y_{CS\text{-}G}$ *and* MK.

Step C3: The user U downloads the client module A on his or her smartphone SP.

Step C4: The user U install the client module A on his or her smartphone SP.

Enrolment Phase

This phase is initiated and performed once when a user requires a connection, access, and control to the IoT devices and their data stored in the cloud server. Fig. 3 illustrates the enrolment phase.

Step E1: The user U run the client module A from his or her smartphone SP.

Step E2: The user U chooses his or her preferred username Id_U and password PW. The client module A generates a shared secret key CK and sends it to the cloud server CS via a secure channel.

Step E3: The cloud server CS generates a user profile for a new user U and stores the secret key CK in an encrypted database.

Step E4: The cloud server CS computer $H_{IdCS} = h(Id_{CS})$ and sends an enrolment respond message $<HId_{CS}>$ to the client module A.

Step E5: The client module A stores the H_{IdCS} in its local storage.

Step E6: The client module A computes $E_{PW} = PW \oplus CK$, and $H_{EPW} = h(PW // Id_{SP})$.

Step E7: The client module A sends a registration request message $<MS_{Id}, T, E_{PW}, H_{EPW}, Id_U, Id_{SP}>$ to the cloud server CS.

Step E8: The cloud server CS receives the registration request message from the client module and decrypts EPW by computing $D_{PW} = (E_{PW} \oplus CK)$.

Step E9: The cloud server CS computes

$H'_{EPW} = h\ (PW\ ||\ Id_{SP})$.

Step E10: The cloud server *CS* verifies the integrity of *PW* by comparing the H'_{EPW} and H_{EPW}.

Step E11: The cloud server *CS* stores H_{EPW}, Id_U, Id_{SP} in an encrypted database.

Step E12: The cloud server *CS* sends a request message for fingerprint registration of user $< MS_{Id}, Id_U, Id_{SP}>$ to the client module *A*.

Step E13: The client module *A* send a fingerprint request message $< MS_{Id}, T, Id_U, F>$ to user *U*.

Step E14: The user *U* activates the fingerprint sensor on the smartphone *SP*.

Step E15: The fingerprint sensor on the smartphone *SP* is activated and ready to capture the user *U*'s fingerprint image. The user *U* may use either his or her thumb or index finger for the enrolment process.

Step E16: The client module *A* captures multiple fingerprint images of the user *U*.

Step E17: The client module *A* converts the user *U's* fingerprint images into a set of digital fingerprint template *F*.

Step E18: The client module A computes $E_F = F \oplus CK$, and $H_{EF} = h\ (E_F\ ||\ F)$.

Step E19: The client module *A* keeps *EF* and *HEF* in its local storage.

Step E20: The client module *A* sends a fingerprint registration request message $< MS_{Id}, T, E_F, H_{EF}, Id_U, Id_{SP}>$ to the cloud server *CS*.

Step E21: The cloud server *CS* receives the fingerprint registration request message from the client module and decrypts *EF* by computing $D_F = (E_F \oplus CK)$.

Step E22: The cloud server CS computes $H'_{EF} = h\ (F\ ||\ Id_{SP})$.

Step E23: The cloud server *CS* verifies the integrity of F by comparing the H'_{EF} and H_{EF}.

Step E24: The cloud server *CS* amends the encrypted database storing the record of Id_U, Id_{SP} with H_{EF} and *F*.

Authentication Phase

This phase is initiated only when a registered user requires access to the IoT devices and their data stored in the cloud server. Fig. 4 illustrates the authentication phase.

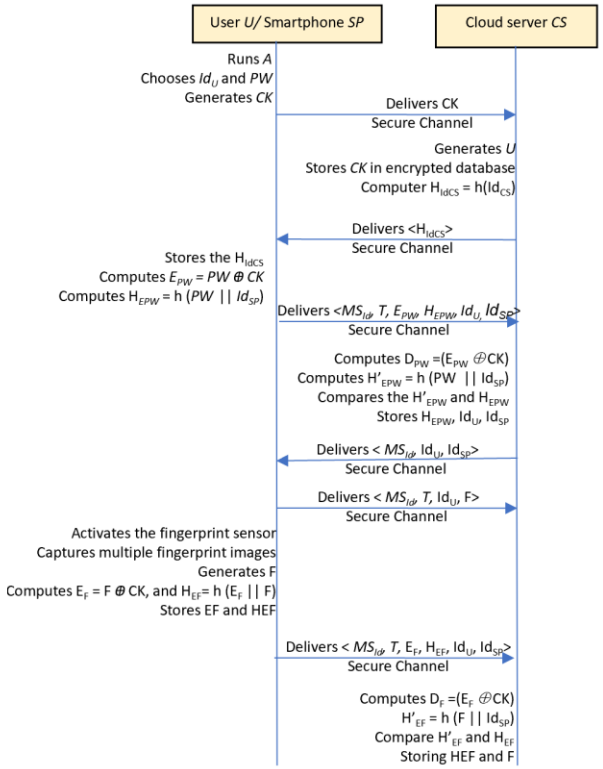

Fig. 3 Enrolment phase.

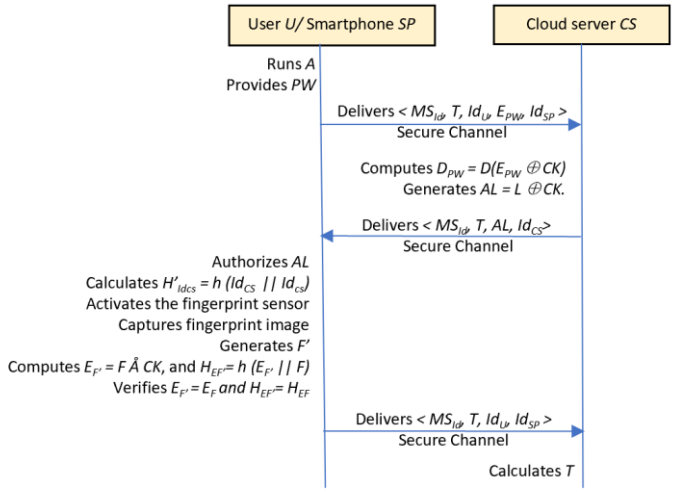

Fig. 4 Authentication phase.

Step A1: The user *U* activates the client module *A* and provides his or her password *PW*.

Step A2: The client module *A* sends an authentication request message $< MS_{Id}, T, Id_U, E_{PW}, Id_{SP} >$.

Step A3: The cloud server *CS* verifies the user *U* identity by computing $D_{PW} = D(E_{PW} \oplus CK)$.

Step A4: The cloud server *CS* generates an encrypted authenticated link *AL* by using an activation link *L* so that $AL = L \oplus CK$.

Step A5: The cloud server *CS* sends a response message < MS_{Id}, *T, AL,* Id_{CS}> for an authenticated activation link *AL* to the user's smartphone *SP*.

Step A6: The user *U* receives the response message and authorizes the link, which reinvokes the client module *A* on the user's smartphone *SP*.

Step A7: The client module *A* verifies the server's authenticity by calculating $H'_{Idcs} = h\ (Id_{CS}\ ||\ CK)$, the *AL* using the shared key *CK*, and the timestamp *T* where the time interval $(T^{*} - T) \geq \Delta T$.

Step A8: The client module *A* activates the fingerprint sensor on the user's smartphone *SP* and is ready to capture the user *U*'s fingerprint image for authentication.

Step A9: The client module *A* converts the user *U*'s fingerprint images into a set of digital fingerprint template *F'*.

Step A10: The client module *A* computes $E_{F'} = F \oplus CK$, and $H_{EF'} = h\ (E_{F'}\ ||\ F)$.

Step A11: The client module *A* verifies that $E_{F'} = E_F$ *and* $H_{EF'} = H_{EF}$.

Step A12: The client module *A* sends an authentication granted message < MS_{Id}, *T,* Id_U, Id_{SP} > *to* the cloud server *CS*.

Step 13: The server *CS* calculates the timestamp *T,* where the time interval $(T^{*} - T) \geq \Delta T$. The user *U* is granted access to data on the cloud server *CS* if the timestamp is valid. Otherwise, the session is considered expired.

Password update phase

This phase is initiated when a registered user requires changing their password for a COT system. The module is invoked only when a registered user U has been authenticated by the cloud server CS with an active session. Fig. 5 illustrates the password update phase.

Step U1: The user *U* activates the client module *A* for password update.

Step U2: The client module *A* sends a password change request message < MS_{Id}, *T,* E_{PW}, H_{EPW}, Id_U, Id_{SP}> to the cloud server *CS*.

Step U3: The cloud server *CS computes* $H'_{EPW} = h\ (PW\ ||\ CK)$ and verifies the integrity of *PW* by comparing the H'_{EPW} and H_{EPW}.

Step U4: The cloud server *CS* computes $H_{IdCS} = h(Id_{CS})$ and sends a password update response message < M_{SId}, *T,* H_{IdCS}> to the client module *A*.

Step U5: The user *U* chooses a new password PW_i.

Step U6: The client module *A* computes $E_{PW} = PWi \oplus CK$, and $H_{EPWi} = h\ (PWi\ ||\ Id_{SP})$.

Step U7: The client module *A* sends a password updated message <MS_{Id}, *T,* E_{PWi}, H_{EPWi}, Id_U, Id_{SP}> to the cloud server *CS*.

Step U8: The cloud server *CS* receives the password updated message from the client module and decrypts E_{PWi} by computing $D_{PWi} = (E_{PWi} \oplus CK)$.

Step U9: The cloud server *CS* computes $H'_{EPWi} = h\ (PWi\ ||\ Id_{SP})$.

Step U10: The cloud server *CS* verifies the integrity of PW_i by comparing the H'_{EPWi} and H_{EPWi}.

Step U11: The cloud server *CS* amends the encrypted database storing the record of Id_U, Id_{SP} with H_{EPWi}.

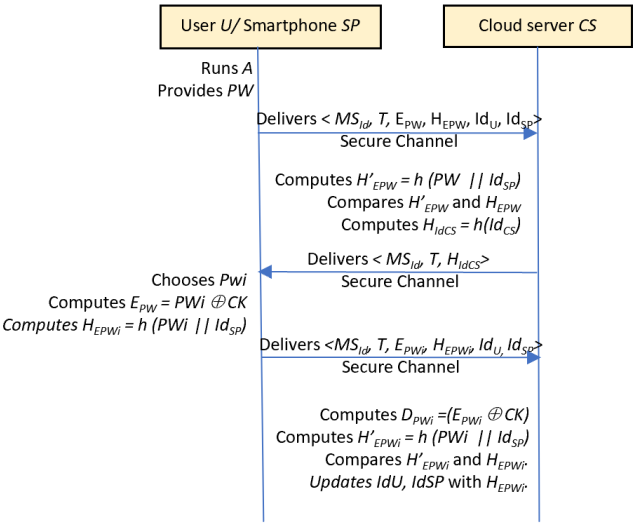

Fig. 5 Password update phase.

Fingerprint revocation phase

This phase is initiated when a registered user requires the cancellation of their fingerprint data for authentication in a COT system. Cancellation of the fingerprint data may be due to injury or damage to the fingerprint skin. Fig. 6 illustrates the fingerprint revocation phase.

Step F1: The user *U* activates the client module *A* for fingerprint revocation.

Step F2: The client module *A* sends a fingerprint revocation request message < MS_{Id}, *T,* E_{PW}, Id_U, Id_{SP}> to the cloud server *CS*.

Step F3: The cloud server *CS* computes $H'_{EPW} = h\ (PW\ ||\ CK)$ and verifies the integrity of *PW* by comparing the H'_{EPW} and H_{EPW}.

Step F4: The cloud server *CS* sends a fingerprint revocation response

message < MS_{Id}, T, H_{IdCS}> to module A in smartphone SP.

Step F5: The user U accepts response message < MS_{Id}, T, H_{IdCS}> on the smartphone SP.

Step F6: The user U generates current timestamp T_i, embeds on revocation message < MS_{Id}, T_i, H_{IdCS}>.

Step F7: The client module A sends fingerprint revocation message <MS_{Id}, T_i, H_{IdCS}> to the cloud server CS.

Step F8: The cloud server CS deletes HEF and F from the database.

Step F9: The client module A deletes HEF and F from its local storage.

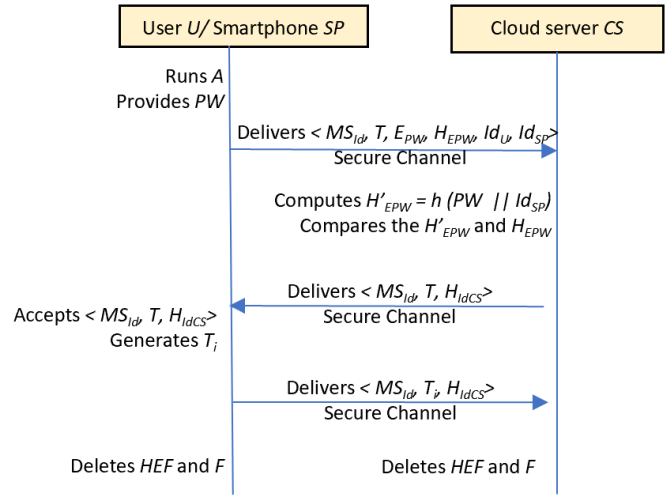

Fig. 6 Fingerprint revocation phase.

Smartphone revocation phase

This phase is initiated when a user changes their smartphone due to a stolen or broken device. Fig. 7 illustrates the smartphone revocation phase.

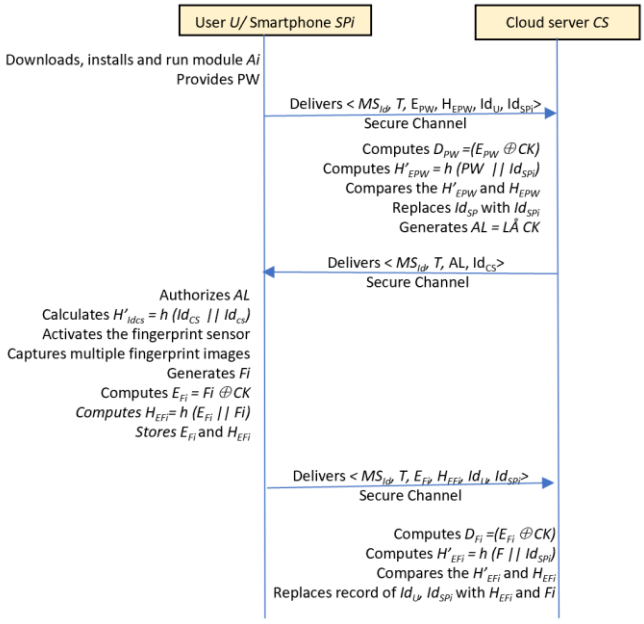

Fig. 7 Smartphone revocation phase.

Step S1: The user U downloads the client module A_i on the new smartphone SP_i.

Step S2: The user U installs the client module A_i on the new smartphone SP_i.

Step S3: The user U runs the client module A_i from the new smartphone SP_i activates and provides the new password PW for the smartphone revocation

Step S4: The client module A_i sends a smartphone revocation request message < MS_{Id}, T, E_{PW}, H_{EPW}, Id_U, Id_{SPi}> to the cloud server CS.

Step S5: The cloud server CS decrypts E_{PW} by computing $D_{PW} = (E_{PW} \oplus CK)$.

Step S6: The cloud server CS computes $H'_{EPW} = h\ (PW\ ||\ Id_{SPi})$ and verifies the integrity of PW by comparing the H'_{EPW} and H_{EPW} and replaces Id_{SP} with Id_{SPi} in the encrypted database storing the record of Id_U.

Step S7: The cloud server CS generates an encrypted authenticated link AL using an activation link L so that $AL = L \oplus CK$.

Step S8: The cloud server CS sends a smartphone revocation response message < MS_{Id}, T, AL, H_{IdCS}> for an authenticated activation link AL to the user's new smartphone SP_i.

Step S9: The user U receives the response message and authorizes the link, which reinvokes the client module A on the users' new smartphone SP_i.

Step S10: The client module A verifies the server's authenticity by calculating $H'_{Idcs} = h\ (IdCS\ ||\ CK)$, the AL using the shared key CK, and the timestamp T where the time interval $(T^* - T) \geq \Delta T$.

Step S11: The client module A activates the fingerprint sensor on the user's smartphone SP_i and is ready to capture the user U's fingerprint image for enrolment.

Step S12: The client module A captures multiple fingerprint images of the user U.

Step S13: The client module A converts the user U's fingerprint images into a set of digital fingerprint template F_i.

Step S14: The client module A computes $E_{Fi} = Fi \oplus CK$, and $H_{EFi} = h\ (E_{Fi}\ ||\ Fi)$.

Step S15: The client module A keeps E_{Fi} and H_{EFi} in its local storage.

Step S16: The client module A sends a fingerprint registration request message < MS_{Id}, T, E_{Fi}, H_{EFi}, Id_U, Id_{SPi}> to the cloud server CS.

Step S17: The cloud server CS receives the

fingerprint registration request message from the client module and decrypts E_{Fi} by computing $D_{Fi} = (E_{Fi} \oplus CK)$.

Step S18: The cloud server *CS* computes $H'_{EFi} = h(F \| Id_{SPi})$.

Step S19: The cloud server *CS* verifies the integrity of *F* by comparing the H'_{EFi} and H_{EFi}.

Step S20: The cloud server *CS* replaces the data in the encrypted database storing the record of Id_U, Id_{SPi} with H_{EFi} and F_i.

4. Evaluation and results

4.1 Security Analysis

The proposed user authentication scheme was analyzed in terms of four types of attacks for COT systems:

1. Denial-of-service attacks: Malicious persons may create traffics to the cloud server *CS*. However, the cloud server CS allows the communication message for user authentication. At the same time, it discards traffics that contain invalid message id MS_{Id} and timestamp *T*. Hence, they could not cause flood and service breakdown at the cloud server.
2. User impersonating attacks: Malicious persons could not use legitimate users' credentials to gain access to COT systems because the user credential communicated during the enrolment and authentication is encrypted. Further, the cloud server authenticates the user and the device used every time for the authentication process through the authentication response message $< MS_{Id}, T, Id_U, E_{PW}, Id_{SP} >$. Every authentication attempt also requires the activation of authenticated link AL sent to the user's smartphone SP. Then, the malicious persons cannot obtain the encrypted fingerprint template stored in the local storage of the smartphone SP because the hash function in $E_{F'} = F \oplus CK$, and $H_{EF'} = h(E_{F'} \| F)$ is irreversible. Therefore, the user impersonating attacks is impossible.
3. Replay attacks: If malicious persons resend the message to the user or the cloud server, the request or response messages would be invalid as the timestamp T does not fulfil the time interval $(T^* - T) \geq \Delta T$.
4. Man-in-the-middle attacks: Malicious persons could not capture the message, manipulate, and resend the message like a legitimate user because users are verified based on the user id and smartphone id. The authentication process required a new and current authenticated link AL. In addition, the users need to capture their fingerprint images for comparison with those stored in the local storage of smartphones. The malicious persons cannot obtain the encrypted fingerprint template stored in the local storage of the smartphone SP because the hash function in $E_{F'} = F \oplus CK$ and $H_{EF'} = h(E_{F'} \| F)$ is irreversible. Therefore, the scheme is secured against man-in-the-middle attacks.

Based on the above analysis, the proposed user authentication scheme is resistant to denial-of-service attacks, user impersonating attacks, replay attacks, and man-in-the-middle attacks. The scheme also fulfils basic user authentication requirements, including mutual authentication, confidentiality, anonymity, and forward secrecy.

1. Mutual authentication: The cloud server and the user embedded their encrypted or hashed identity in the request and response message like $< MS_{Id}, T, H_{IdCS} >$ and $< MS_{Id}, T, E_{PW}, Id_U, Id_{SP} >$. Hence, each party authenticated themselves during their communication.
2. Confidentiality: The use of concatenation, hash and XOR operations on the data protected their confidentiality.
3. Anonymity: Malicious persons could not capture the data as the communication channels are assumed to be secured.
4. Forward secrecy: The secret key used in this scheme is communicated using a secure channel where it is not known to the malicious persons.

4.2 Evaluation Using ProVerif

ProVerif [51-60] is an automated tool for verifying security schemes and protocols [49, 61]. Many recent studies like [62-66] also used Proverif as a tool to verify their authentication schemes. The protocols involved in the proposed user authentication scheme were transformed into ProVerif code. Each phase in the proposed scheme was verified individually, involving different threats and attacks. In ProVerif, the communication channels and the data communicated between the communicating entities should be declared or defined similarly to writing codes in any programming language.

First, the ProVerif code evaluated the configuration phase that involved two communication channels, commIG and CommGCS, which are intended for communication between the IoT devices with the gateway and the gateway with the cloud server. *IdI, IdCS, MK*, and *SK* were defined as four constants representing the IoT devices' unique identifier, the cloud server, the secret key, and the shared key, respectively, as shown in Fig. 8. Two functions were involved in this phase, namely Concatenation and Hash. Two events were also declared to indicate the beginning and end of the phase. The queries defined in the code intend to check whether the unique IoT devices, cloud server

identifiers, and secret and shared keys were known to the attackers. The summary of the results for the verification is illustrated in Fig. 9. The results suggested that the attackers cannot get the unique identifiers and keys communicated during the configuration phase.

```
(*----CONFIGURATION PHASE----*)
(*Secure Communication Channel*)
free commIG:channel [private].
free commGCS:channel [private].
(*Identity of the communicating entities*)
free IdI:bitstring [private].
free IdCS:bitstring [private].
(*Secret Key *)
free MK:bitstring [private].
(*Shared key*)
free SK:bitstring [private].
(*Functions*)
fun Concatenation(bitstring,
bitstring):bitstring.
fun Hash(bitstring):bitstring.
(*Events*)
event StartConfiguration(bitstring).
event EndConfiguration(bitstring).
(*Checking whether the data are known to the
attacker*)
query attacker(IdI).
query attacker(IdCS).
query attacker(MK).
query attacker(SK).
Query id:bitstring;inj-
event(EndConfiguration(IdCS))==>inj-
event(StartConfiguration(IdI)).
process
event StartConfiguration(IdI);
let YGIi=Hash(Concatenation(IdI,MK)) in
let YCSG=Hash(Concatenation(IdCS,SK)) in
out(commGCS,YCSG);
event EndConfiguration(IdCS);
0
```

Fig. 8 ProVerif code for verifying the configuration phase.

```
----------------------------------------
Verification summary:
Query not attacker(IdI[]) is true.
Query not attacker(IdCS[]) is true.
Query not attacker(MK[]) is true.
Query not attacker(SK[]) is true.
Query inj-event(EndConfiguration(IdCS[]))
==> inj-event(StartConfiguration(IdI[])) is
true.
```

Fig. 9 Verification results for the configuration phase.

In the enrolment phase, two other communication channels were used, namely commSP and CommCS, representing the initiation of communication for the user and the cloud server, respectively. These two communication channels will be used for the other subsequent phases of the scheme. Six constants were defined, representing the unique identifier for the user, the cloud server, the smartphone, and the secret key. Other constants include the fingerprint template and the authenticated link. Concatenation and Hash functions remain with an additional XOR1 function for performing the XOR operations. Seven events were also declared, representing enrolment activities. The queries check whether or not all information involved in the phase was known to the attackers. The complete Proverif code is presented in Fig. 10. The summary of the results for the verification of the enrolment phase is illustrated in Fig. 11. The results suggested that the attackers cannot get the information during the configuration phase

```
(*----ENROLMENT PHASE----*)
(*Secure Communication Channel*)
free commSPCS:channel[private].
free commCSSP:channel[private].
(*Identity of the communicating entities*)
free IdU:bitstring[private].
free PW:bitstring[private].
free IdSP:bitstring[private].
free IdCS:bitstring[private].
free F:bitstring[private].
free L:bitstring[private].
(*Secret Key*)
free CK:bitstring [private].
(*Functions*)
fun
Concatenation(bitstring,bitstring):bitstring
.
fun Hash(bitstring):bitstring.
fun XOR1(bitstring,bitstring):bitstring.
(*Events*)
event
StartAuthentication(bitstring,bitstring,bits
tring).
event SuccessfulAuthentication.
event UnSuccessfulAuthentication.
event GenerateAuthenticationLink.
event ActivateSensor.
event FingerprintTemplate(bitstring).
event
EndAuthentication(bitstring,bitstring,bitstr
ing).
(*Checking whether the data are known to the
attacker*)
query attacker(IdU).
query attacker(PW).
query attacker(CK).
query attacker(IdSP).
query attacker(IdCS).
query attacker(F).
query attacker(L).
query id:bitstring;inj-
event(EndAuthentication(IdU,PW,IdSP))==>inj-
event(StartAuthentication(IdU,PW,IdSP)).
process
event StartAuthentication(IdU,PW,IdSP);
new EPW:bitstring;
new MSId:bitstring;
new T:bitstring;
new DPW:bitstring;
new AL:bitstring;
new HIdCS:bitstring;
new H1IdCS:bitstring;
new T1:bitstring;
new F1:bitstring;
new EF:bitstring;
new EF1:bitstring;
new HEF:bitstring;
new HEF1:bitstring;
let EPW=XOR1(PW,CK) in
out(commSPCS,(MSId,T,EPW,IdSP));
let DPW=XOR1(PW,CK) in
if DPW=EPW then event
GenerateAuthenticationLink else event
UnSuccessfulAuthentication;
let AL=XOR1(L,CK) in
out(commCSSP,(MSId, T, AL, IdCS));
let HIdCS=Hash(Concatenation(IdCS,CK)) in
let H1IdCS=Hash(Concatenation(IdCS,CK)) in
if HIdCS=H1IdCS then event ActivateSensor
else event UnSuccessfulAuthentication;
event FingerprintTemplate(F1);
let EF=XOR1(F,CK) in
let HEF=Hash(XOR1(F,CK)) in
```

```
let EF1=XOR1(F,CK) in
let HEF1=Hash(XOR1(F,CK)) in
if HEF1=HEF then
out(commSPCS,(MSId,T,IdU,IdSP))else event
UnSuccessfulAuthentication;
if T<>T1 then event SuccessfulAuthentication
else event UnSuccessfulAuthentication;
event EndAuthentication(IdU,PW,IdSP);
0
```

Fig. 10 Fraction of ProVerif code for verifying the enrolment phase

```
----------------------------------------
Verification summary:
Query not attacker(IdU[]) is true.
Query not attacker(PW[]) is true.
Query not attacker(CK[]) is true.
Query not attacker(IdSP[]) is true.
Query not attacker(IdCS[]) is true.
Query not attacker(F[]) is true.
Query not attacker(L[]) is true.
Query inj-
event(EndAuthentication(IdU[],PW[],IdSP[]))
==> inj-
event(StartAuthentication(IdU[],PW[],IdSP[])
) is true.
```

Fig. 11 Verification results for the enrolment phase.

The Proverif code for the authentication phase is illustrated in Fig. 12. In this phase, the channels' declaration, constants, functions, events, and queries are similar to the enrolment phase. The only difference is the process involved in the authentication phase. The results of the verification of the authentication phase are summarized in Fig. 13.

```
(*----AUTHENTICATION PHASE----*)
(*Secure Communication Channel*)
free commSPCS:channel[private].
free commCSSP:channel[private].
(*Identity of the communicating entities*)
free IdU:bitstring[private].
free PW:bitstring[private].
free IdSP:bitstring[private].
free IdCS:bitstring[private].
free F:bitstring[private].
free L:bitstring[private].
(*Secret Key*)
free CK:bitstring [private].
(*Functions*)
fun XOR1(bitstring,bitstring):bitstring.
(*Events*)
event
StartAuthentication(bitstring,bitstring,bits
tring).
event UnSuccessfulAuthentication.
event GenerateAuthenticationLink.
event
EndAuthentication(bitstring,bitstring,bitstr
ing).
(*Checking whether the data are known to the
attacker*)
query attacker(IdU).
query attacker(PW).
query attacker(CK).
query attacker(IdSP).
query attacker(IdCS).
query attacker(F).
query attacker(L).
query id:bitstring;inj-
event(EndAuthentication(IdU,PW,IdSP))==>inj-
event(StartAuthentication(IdU,PW,IdSP)).
process
event StartAuthentication(IdU,PW,IdSP);
new EPW:bitstring;
new MSId:bitstring;
new T:bitstring;
```

```
new DPW:bitstring;
let EPW=XOR1(PW,CK) in
out(commSPCS,(MSId,T,EPW,IdSP));
let DPW=XOR1(PW,CK) in
if DPW=EPW then event
GenerateAuthenticationLink else event
UnSuccessfulAuthentication;
event EndAuthentication(IdU,PW,IdSP);
0
```

Fig. 12 ProVerif code for verifying the authentication phase.

```
----------------------------------------
Verification summary:
Query not attacker(IdU[]) is true.
Query not attacker(PW[]) is true.
Query not attacker(CK[]) is true.
Query not attacker(IdSP[]) is true.
Query not attacker(IdCS[]) is true.
Query not attacker(F[]) is true.
Query not attacker(L[]) is true.
Query inj-
event(EndAuthentication(IdU[],PW[],IdSP[]))
==> inj-
event(StartAuthentication(IdU[],PW[],IdSP[])
) is true.
```

Fig. 13 Verification results for the authentication phase.

The verification process was also done in the remaining phases covering the password update, fingerprint revocation, and smartphone revocation phases. Only the results of the Proverif verifications are presented due to the lengthy code. The results for password update, fingerprint revocation, and smartphone revocation phases are illustrated in Fig. 14, Fig. 15, and Fig. 16, respectively.

```
----------------------------------------
Verification summary:
Query not attacker(IdU[]) is true.
Query not attacker(PW[]) is true.
Query not attacker(CK[]) is true.
Query not attacker(IdSP[]) is true.
Query not attacker(IdCS[]) is true.
Query not attacker(PWi[]) is true.
Query inj-
event(EndPWUpdate(IdU[],PW[],IdSP[])) ==>
inj-event(StartPWUpdate(IdU[],PWi[],IdSP[]))
is true.
```

Fig. 14 Verification results for the password update phase.

```
----------------------------------------
Verification summary:
Query not attacker(IdU[]) is true.
Query not attacker(PW[]) is true.
Query not attacker(CK[]) is true.
Query not attacker(IdSP[]) is true.
Query not attacker(IdCS[]) is true.
Query inj-
event(EndFPRevocation(IdU[],PW[],IdSP[]))
==> inj-
event(StartFPRevocation(IdU[],PW[],IdSP[]))
is true.
```

Fig. 15 Verification results for the fingerprint revocation phase.

```
----------------------------------------
Verification summary:
Query not attacker(IdU[]) is true.
Query not attacker(PW[]) is true.
Query not attacker(CK[]) is true.
Query not attacker(IdSPi[]) is true.
Query not attacker(IdCS[]) is true.
Query not attacker(L[]) is true.
Query inj-
```

```
event(EndSPRevocation(IdU[],PW[],IdSPi[]))
==> inj-
event(StartSPRevocation(IdU[],PW[],IdSPi[]))
is true.
```

Fig. 16 Verification results for the smartphone revocation phase.

The proposed authentication scheme has been evaluated through security analysis and ProVerif automated verification. The analysis suggested that the scheme is secure and suitable to be used within COT. The outcome of this study would be able to support the need for securing the entire communication of such systems. IoT, cloud, and smartphones are expected to grow exponentially to support various smart and intelligent systems; thus, authentication is considered an essential requirement for IoT [18, 19]. Further, biometric authentication, like a fingerprint, has higher security than alphanumeric-based passwords [25]. Further, they can be used with a password and email link to form a multi-factor authentication that provides robust security and efficiency for the IoT environment [26].

5. Conclusion and Future Works

This study proposed a user authentication scheme for COT that intends to meet the security demands to secure communication between a remote server and user for accessing the IoT data through the mobile cloud computing environment. The proposed authentication comprises six phases: configuration, enrolment, authentication, password update, fingerprint revocation, and smartphone revocation. This authentication scheme utilizes the fingerprint sensor on the smartphone to accomplish the authentication process to access data of IoT-based applications stored on a cloud server. Also, it uses a one-time authentication link sent by the cloud server to the user's smartphone to strengthen the communication. The scheme used XOR and hash for data encryption and integrity protection and was evaluated using an automated verification tool. The automated and informal evaluation demonstrated that the proposed scheme fulfils the essential requirement for remote user authentication schemes and resistance against known attacks.

Competing interests

The authors have declared that no competing interests exist.

Funding

The Ministry of Higher Education Malaysia funded this study under the Fundamental Research Grant Scheme (Ref: FRGS/1/2018/ICT03/UUM/02/1, UUM S/O Code: 14208).

Authors' contribution

All the authors in this document participated in the development of and successful completion of the research. All authors read and approved the final manuscript.

References

[1] N. N. Mohamed, Y. M. Yussoff, M. A. Saleh, and H. Hashim, "Hybrid cryptographic approach for Internet of things applications: A review," *Journal of Information and Communication Technology,* vol. 19, no. 3, pp. 279-319, 2020.

[2] I. U. Din, M. Guizani, J. J. P. C. Rodrigues, S. Hassan, and V. V. Korotaev, "Machine learning in the Internet of Things: Designed techniques for smart cities," *Future Generation Computer Systems,* vol. 100, no. November, pp. 826-843, 2019.

[3] K. Ahmad, O. Mohammad, M. Atieh, and H. Ramadan, "Enhanced performance and faster response using new IoT litetechnique," *International Arab Journal of Information Technology,* vol. 16, no. 3A, pp. 548-556, 2019.

[4] B. D. Deebak and F. Al-Turjman, "Lightweight authentication for IoT/Cloud-based forensics in intelligent data computing," *Future Generation Computer Systems,* vol. 116, no. March, pp. 406-425, 2021.

[5] A. M. Rashid, A. A. Yassin, A. A. A. Wahed, and A. J. Yassin, "Smart city security: Face-based image retrieval model using gray level cooccurrence matrix," *Journal of Information and Communication Technology,* vol. 19, no. 3, pp. 437-458, 2020.

[6] C.-C. Chang, H.-L. Wu, and C.-Y. Sun, "Notes on "Secure authentication scheme for IoT and cloud servers"," *Pervasive and Mobile Computing,* vol. 38, no. July, pp. 275-278, 2017.

[7] N. Katuk, K. R. Ku-Mahamud, N. H. Zakaria, and M. A. Maarof, "Implementation and recent progress in cloud-based smart home automation systems," in *ISCAIE 2018 - 2018 IEEE Symposium on Computer Applications and Industrial Electronics*, 2018, pp. 71-77.

[8] B. Guo, D. Zhang, Z. Wang, Z. Yu, and X. Zhou, "Opportunistic IoT: Exploring the harmonious interaction between human and the Internet of Things," *Journal of Network and Computer Applications,* vol. 36, no. 6, pp. 1531-1539, 2013.

[9] F. Alhaidari, A. Rahman, and R. Zagrouba, "Cloud of Things: architecture, applications and challenges," *Journal of Ambient Intelligence and Humanized Computing,* 2020.

[10] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of Cloud computing and Internet of Things: A survey," *Future Generation Computer Systems,* vol. 56, no. March, pp. 684-700, 2016.

[11] F. Daneshgar, O. A. Sianaki, and A. Ilyas, "Overcoming Data Security Challenges of Cloud of Things: An Architectural Perspective," *Advances in Intelligent Systems and Computing,* vol. 993, pp. 646-659, 2020.

[12] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Integration of Blockchain and Cloud of Things: Architecture, Applications and Challenges," *IEEE Communications Surveys and Tutorials,* vol. 22, no. 4, pp. 2521-2549, 2020.

[13] T. C. S. Xavier, I. L. Santos, F. C. Delicato, P. F. Pires, M. P. Alves, T. S. Calmon*, et al.*, "Collaborative resource allocation for Cloud of Things systems," *Journal of Network and Computer Applications,* vol. 159, no. June, pp. 102592, 2020.

[14] S. Xuan and D. H. Kim, "Development of Cloud of Things Based on Proxy Using OCF IoTivity and MQTT for P2P Internetworking," *Peer-to-Peer Networking and Applications,* vol. 13, no. 3, pp. 729-741, 2020.

[15] H. Elazhary, "Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: Disambiguation and research directions," *Journal of Network and Computer Applications,* vol. 128, no. February, pp. 105-140, 2019.

[16] H. Nguyen, H. H. Nguyen, T. Hoang, D. Choi, and T. D. Nguyen, "A Generalized Authentication Scheme for Mobile Phones Using Gait Signals," in *International Conference on E-Business and Telecommunications*, 2015, pp. 386-407.

[17] A. Ayoub, R. Najat, and A. Jaafar, "A lightweight secure CoAP for IoT-cloud paradigm using elliptic-curve cryptography," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 20, no. 3, pp. 1460-1470, 2020.

[18] H. L. Wu, C. C. Chang, Y. Z. Zheng, L. S. Chen, and C. C. Chen, "A secure IoT-based authentication system in cloud computing environment," *Sensors (Switzerland),* vol. 20, no. 19, pp. 1-14, 2020.

[19] Z. Houhamdi and B. Athamena, "Identity identification and management in the internet of things," *International Arab Journal of Information Technology,* vol. 17, no. 44, pp. 645-654, 2020.

[20] S. M. Kannan Mani, M. Balaji Bharatwaj, and N. Harini, "A Scheme to Enhance the Security and Efficiency of MQTT Protocol," *Smart Innovation, Systems and Technologies,* vol. 194, pp. 79-93, 2021.

[21] B. Liao, Y. Ali, S. Nazir, L. He, and H. U. Khan, "Security Analysis of IoT Devices by Using Mobile Computing: A Systematic Literature Review," *IEEE Access,* vol. 8, pp. 120331-120350, 2020.

[22] N. Katuk, N. H. Zakaria, and K. R. Ku-Mahamud, "Mobile phone sensing using the built-in camera," *International Journal of Interactive Mobile Technologies,* vol. 13, no. 2, pp. 102-114, 2019.

[23] I. A. Alnajjar and M. Mahmuddin, "Feature indexing and search optimization for enhancing the forensic analysis of mobile cloud environment," *Information Security Journal,* vol. 30, no. 4, pp. 235-256, 2020.

[24] H. B. Alwan and K. R. Ku-Mahamud, "Cancellable face template algorithm based on speeded-up robust features and winner-takes-all," *Multimedia Tools and Applications,* vol. 79, pp. 28675-28693, 2020.

[25] P. C. Venugopal and K. S. A. Viji, "Applying empirical thresholding algorithm for a keystroke dynamics based authentication system," *Journal of Information and Communication Technology,* vol. 18, no. 4, pp. 383-413, 2019.

[26] S. Desai Karanam, S. Shetty, and K. U. G. Nithin, "Fog computing application for biometric-based secure access to healthcare data," *Signals and Communication Technology,* pp. 355-383, 2021.

[27] K. Shanmugasundaram, A. S. A. Mohmed, and N. I. R. Ruhaiyem, "Hybrid improved bacterial swarm optimization algorithm in hand-based multimodal biometric authentication system," *Journal of Information and Communication Technology,* vol. 18, no. 2, pp. 123-141, 2019.

[28] A. Siswanto, N. Katuk, and K. R. Ku-Mahamud, "Chaotic-based encryption algorithm using henon and logistic maps for fingerprint template protection," *International Journal of Communication Networks and Information Security,* vol. 12, no. 1, pp. 1-9, 2020.

[29] I. Velásquez, A. Caro, and A. Rodríguez, "Kontun: A Framework for recommendation of authentication schemes and methods," *Information and Software Technology,* vol. 96, no. April, pp. 27-37, 2018.

[30] M. Masdari and S. Ahmadzadeh, "A survey and taxonomy of the authentication schemes in Telecare Medicine Information Systems," *Journal of Network and Computer Applications,* vol. 87, no. June, pp. 1-19, 2017.

[31] S. Asha and C. Chellappan, "Authentication of e-learners using multimodal biometric technology," in *International Symposium on Biometrics and Security Technologies, 2008. ISBAST 2008,* 2008, pp. 1-6.

[32] S. Seno, T. Sadakane, Y. Baba, T. Shikama, Y. Koui, and N. Nakaya, "A network authentication system with multi-biometrics," in *Communications, 2003. APCC 2003. The 9th Asia-Pacific Conference on*, 2003, pp. 914-918.

[33] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, and Y. Koucheryavy, "Multi-Factor Authentication: A Survey," *Cryptography,* vol. 2, no. 1, pp. 1, 2018.

[34] SAP. (n.d.). *Authentication Scheme*. Available: https://help.sap.com/doc/7ba199e10fdc488293db33f0709f9225/7.5.6/en-US/9052c43dac1bcf51e10000000a114084.html

[35] P. K. Dhillon and S. Kalra, "A lightweight biometrics based remote user authentication scheme for IoT services," *Journal of Information Security and Applications,* vol. 34, no. June, pp. 255-270, 2017.

[36] H. Zhu, J. Hu, S. Chang, and L. Lu, "ShakeIn: Secure User Authentication of Smartphones with Single-Handed Shakes," *IEEE Transactions on Mobile Computing,* vol. 16, no. 10, pp. 2901-2912, 2017.

[37] A. De Santis, M. Flores, and B. Masucci, "One-Message Unilateral Entity Authentication Schemes," presented at the Proceedings of the 12th International Conference on Availability, Reliability and Security, Reggio Calabria, Italy, 2017, pp. 1-6.

[38] I.-E. Liao, C.-C. Lee, and M.-S. Hwang, "A password authentication scheme over insecure networks," *Journal of Computer and System Sciences,* vol. 72, no. 4, pp. 727-740, 2006.

[39] T. Sahayini and M. Manikandan, "Enhancing the security of modern ICT systems with multimodal biometric cryptosystem and continuous user authentication," *International Journal of Information and Computer Security,* vol. 8, no. 1, pp. 55-71, 2016.

[40] D. Dasgupta, A. Roy, and A. Nag, "Multi-Factor Authentication," in *Advances in User Authentication*, Springer, 2017, pp. 185-233.

[41] M. J. Dillon, "Factors that influence adoption of multi-factored authentication within large organizations," PhD Dissertation, Capella University, 2015.

[42] S. H. Khan, M. A. Akbar, F. Shahzad, M. Farooq, and Z. Khan, "Secure biometric template generation for multi-factor authentication," *Pattern Recognition,* vol. 48, no. 2, pp. 458-472, 2015.

[43] D. Dasgupta, A. Roy, and A. Nag, "Toward the design of adaptive selection strategies for multi-factor authentication," *Computers & Security,* vol. 63, no.

November, pp. 85-116, 2016.
[44] S. Kalra and S. K. Sood, "Secure authentication scheme for IoT and cloud servers," *Pervasive and Mobile Computing,* vol. 24, no. December, pp. 210-223, 2015.
[45] P. Gope, R. Amin, S. K. Hafizul Islam, N. Kumar, and V. K. Bhalla, "Lightweight and privacy-preserving RFID authentication scheme for distributed IoT infrastructure with secure localization services for smart city environment," *Future Generation Computer Systems,* vol. 83, no. June, pp. 629-637, 2017.
[46] F. Wu, L. Xu, S. Kumari, X. Li, J. Shen, K.-K. R. Choo*, et al.*, "An efficient authentication and key agreement scheme for multi-gateway wireless sensor networks in IoT deployment," *Journal of Network and Computer Applications,* vol. 89, no. July, pp. 72-85, 2017.
[47] R. Amin, N. Kumar, G. P. Biswas, R. Iqbal, and V. Chang, "A light weight authentication protocol for IoT-enabled devices in distributed Cloud Computing environment," *Future Generation Computer Systems,* vol. 78, no. January, pp. 1005-1019, 2018.
[48] T. Maitra, M. S. Obaidat, R. Amin, S. Islam, S. A. Chaudhry, and D. Giri, "A robust ElGamal-based password-authentication protocol using smart card for client-server communication," *International Journal of Communication Systems,* vol. 30, no. 11, p. e3242, 2017.
[49] Y. Yu, L. Hu, and J. Chu, "A secure authentication and key agreement scheme for IoT-based cloud computing environment," *Symmetry,* vol. 12, no. 1, pp. 150, 2020.
[50] D. He, S. Zeadally, N. Kumar, and W. Wu, "Efficient and Anonymous Mobile User Authentication Protocol Using Self-Certified Public Key Cryptography for Multi-Server Architectures," *IEEE Transactions on Information Forensics and Security,* vol. 11, no. 9, pp. 2052-2064, 2016.
[51] M. Abadi, B. Blanchet, and C. Fournet, "The applied Pi calculus: Mobile values, new names, and secure communication," *Journal of the ACM,* vol. 65, no. 1, pp. 1-41, 2017.
[52] X. Allamigeon and B. Blanchet, "Reconstruction of attacks against cryptographic protocols," in *Proceedings of the Computer Security Foundations Workshop*, 2005, pp. 140-154.
[53] K. Bhargavan, B. Blanchet, and N. Kobeissi, "Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate," in *Proceedings - IEEE Symposium on Security and Privacy*, 2017, pp. 483-502.
[54] B. Blanchet, "Using horn clauses for analyzing security protocols," *Cryptology and Information Security Series,* vol. 5, pp. 86-111, 2011.
[55] B. Blanchet, "Security protocol verification: Symbolic and computational models," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* vol. 7215 LNCS, pp. 3-29, 2012.
[56] B. Blanchet, "Automatic Verification of Security Protocols in the Symbolic Model: The Verifier Proverif," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* vol. 8604, pp. 54-87, 2014.
[57] B. Blanchet, M. Abadi, and C. Fournet, "Automated verification of selected equivalences for security protocols," in *Proceedings - Symposium on Logic in Computer Science*, 2005, pp. 331-340.
[58] B. Blanchet, M. Abadi, and C. Fournet, "Automated verification of selected equivalences for security protocols," *Journal of Logic and Algebraic Programming,* vol. 75, no. 1, pp. 3-51, 2008.
[59] B. Blanchet and A. Chaudhuri, "Automated formal analysis of a protocol for secure file sharing on untrusted storage," in *Proceedings - IEEE Symposium on Security and Privacy*, 2008, pp. 417-431.
[60] V. Cheval and B. Blanchet, "Proving more observational equivalences with ProVerif," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* vol. 7796 LNCS, pp. 226-246, 2013.
[61] M. J. Bhuva and S. Singh, "Symmetric key-based authenticated encryption protocol," *Information Security Journal,* vol. 28, no. 1-2, pp. 35-45, 2019.
[62] C. Dai and Z. Xu, "A secure three-factor authentication scheme for multi-gateway wireless sensor networks based on elliptic curve cryptography," *Ad Hoc Networks,* vol. 127, no. March, pp. 102768, 2022.
[63] M. Xu, G. Xu, H. Xu, J. Zhou, and S. Li, "A decentralized lightweight authentication protocol under blockchain," *Concurrency and Computation: Practice and Experience,* vol. 34, no. 13, pp. e6920, 2022.
[64] M. Ebrahimi, M. Bayat, and B. Zahednejad, "A Privacy Preserving Mutual Authentication Scheme Suitable for IoT-Based Medical Systems," *ISeCure,* vol. 14, no. 1, pp. 57-68, 2022.
[65] M. Bhattacharya, S. Roy, A. K. Das, S. Chattopadhyay, S. Banerjee, and A. Mitra, "DDoS attack resisting authentication protocol for mobile based online social network applications," *Journal of Information Security and Applications,* vol. 65, no. March, pp. 103115, 2022.
[66] B. Hu, W. Tang, and Q. Xie, "A two-factor security authentication scheme for wireless sensor networks in IoT environments," *Neurocomputing,* vol. 500, no. August, pp. 741-749, 2022.

Citation: N. Katuk, R. Vergallo, T. Sugiharto and R. A. Krisdiawan. *A Client-based User Authentication Scheme for the Cloud of Things Environment.* Journal of Computer Science & Technology, vol. 22, no. 2, pp. 102-115, 2022.
DOI: 10.24215/16666038.22.e08
Received: July 18, 2022 **Accepted:** September 5, 2022.