Heliyon 10 (2024) e37577

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Securing the IoT-enabled smart healthcare system: A PUF-based resource-efficient authentication mechanism

Omar Alruwaili [a], Muhammad Tanveer [b], Faisal Mohammed Alotaibi [c], Waleed Abdelfattah [d], Ammar Armghan [e,*], Faeiz M. Alserhani [f]

[a] Department of Computer Engineering and Networks, College of Computer and Information Sciences, Jouf University, Sakaka, 72388, Saudi Arabia
[b] Department of Computer Science, University of Management and Technology, Lahore, Pakistan
[c] Department of Computer Science, Prince Sattam bin Abdulaziz University, Al-Kharj, Ar Riyadh, Saudi Arabia
[d] General Subject Department, University of Business and Technology, Jeddah 23435, Saudi Arabia
[e] Department of Electrical Engineering, College of Engineering, Jouf University, Sakaka 72388, Saudi Arabia
[f] Department of Computer Engineering and Networks, College of Computer and Information Sciences, Jouf University, Sakaka, 72388, Saudi Arabia

ARTICLE INFO

Keywords:
Authentication
Encryption
Security
Healthcare
Scyther
Internet of things

ABSTRACT

As the Internet of Things (IoT) continues its rapid expansion, cloud computing has become integral to various smart healthcare applications. However, the proliferation of digital health services raises significant concerns regarding security and data privacy, making the protection of sensitive medical information paramount. To effectively tackle these challenges, it is crucial to establish resilient network infrastructure and data storage systems capable of defending against malicious entities and permitting access exclusively to authorized users. This requires the deployment of a robust authentication mechanism, wherein medical IoT devices, users (such as doctors or nurses), and servers undergo registration with a trusted authority. The process entails users retrieving data from the cloud server, while IoT devices collect patient data. Before granting access to data retrieval or storage, the cloud server verifies the authenticity of both the user and the IoT device, ensuring secure and authorized interactions within the system. With millions of interconnected smart medical IoT devices autonomously gathering and analyzing vital patient data, the importance of robust security measures becomes increasingly evident. Standard security protocols are fundamental in fortifying smart healthcare applications against potential threats. To confront these issues, this paper introduces a secure and resource-efficient cloud-enabled authentication mechanism. Through empirical analysis, it is demonstrated that our authentication mechanism effectively reduces computational and communication overheads, thereby improving overall system efficiency. Furthermore, both informal and formal analyses affirm the mechanism's resilience against potential cyberattacks, highlighting its effectiveness in safeguarding smart healthcare applications.

* Corresponding author.
E-mail addresses: Oalruwaili@ju.edu.sa (O. Alruwaili), tanveer123giki@gmail.com (M. Tanveer), w.abdelfattah@ubt.edu.sa (W. Abdelfattah), aarmghan@ju.edu.sa (A. Armghan), fmserhani@ju.edu.sa (F.M. Alserhani).

https://doi.org/10.1016/j.heliyon.2024.e37577
Received 29 June 2024; Received in revised form 3 September 2024; Accepted 5 September 2024

Available online 10 September 2024

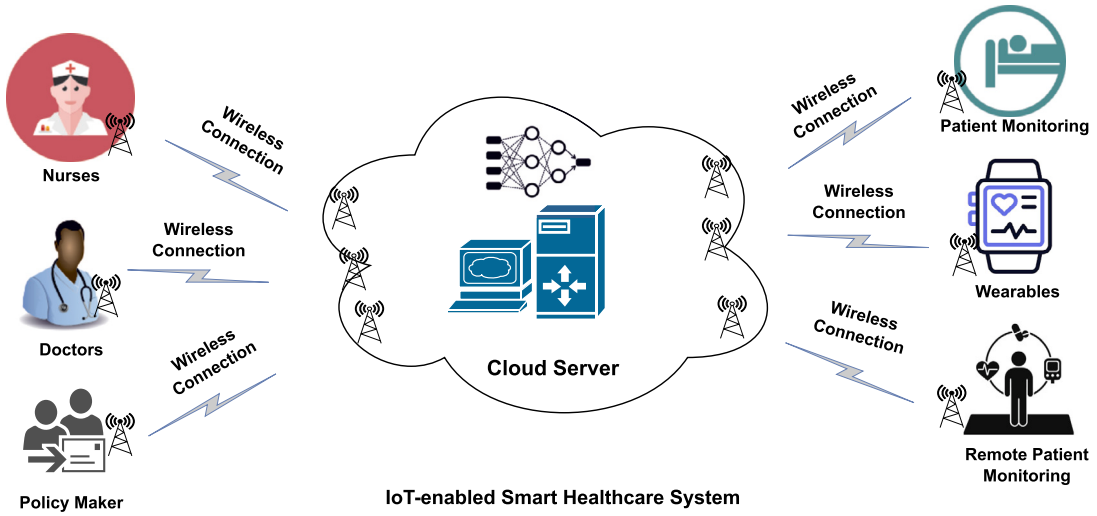


Fig. 1. A use case of SHS: Users receive processed data while IoT devices transmit patient information to cloud servers.

1. Introduction

The Internet of Things (IoT) and intelligent/smart healthcare application represent cutting-edge domains at the intersection of technology and healthcare, revolutionizing the way medical services are delivered and experienced [1–3]. IoT's integration into various sectors has ushered in a new era of connectivity, data analytics, and operational efficiency. IoT promotes smart interaction between devices and systems, allowing businesses to gather, manage, and evaluate large volumes of data instantly. This leads to enhanced decision-making and increased operational flexibility [4,5]. The emergence of IoT-enabled smart healthcare system marks a significant paradigm shift in healthcare delivery, leveraging innovative technologies such as wearable sensors, IoT devices, artificial intelligence, and big data analytics. These systems aim to provide personalized care outside traditional healthcare settings, empowering patients and healthcare providers alike. Through continuous data collection and analysis, IoT-enabled SHS facilitates early disease detection, personalized medical interventions, and overall health improvement [6].

The core of IoT-enabled smart healthcare systems is the effortless communication of health-related data gathered from diverse IoT devices such as wearable health trackers, medical equipment, and monitoring sensors [7,8]. These devices capture real-time health metrics such as vital signs, medication adherence, and physical activity, transmitting this data to centralized servers or cloud platforms for analysis and storage. Fig. 1 shows the smart healthcare system. Integration with artificial intelligence further enhances these systems' capabilities, enabling automated diagnosis and personalized medical recommendations. Despite the transformative potential of IoT-enabled SHS, ensuring the security and privacy of patient data remains a critical challenge [9,10]. The open nature of communication channels used in these systems raises concerns about unauthorized access and data breaches. Malicious actors could exploit vulnerabilities to manipulate or forge medical data, posing significant risks to patient safety and confidentiality [11].

Tackling these challenges necessitates strong authentication and key exchange (AKA) mechanisms to guarantee secure communication between users and medical servers. While traditional cryptographic techniques like symmetric and asymmetric encryption have been utilized in various AKA frameworks, many current solutions suffer from computational inefficiencies and security vulnerabilities, particularly in time-sensitive healthcare settings [12]. To counter this, our paper proposes two tailored AKA mechanisms specifically for IoT-enabled smart healthcare applications. The first enables IoT devices deployed in these applications to retrieve and store data on cloud servers, while the second facilitates users such as doctors and nurses in accessing the stored data. Both mechanisms leverage symmetric encryption algorithm (AES-CBC) and physical unclonable functions (PUFs) to enhance security and reliability while minimizing computational overhead. Through the integration of efficient encryption algorithms and hardware-based security mechanisms, our framework aims to mitigate the inherent security risks in IoT-enabled healthcare applications, ensuring the confidentiality and integrity of patient data.

2. Related work

To ensure seamless communication within IoT-based healthcare systems, numerous AKA mechanisms have been proposed in the exiting literature. In this direction, in [13], the authors developed an AKA mechanism for IoT-based e-healthcare systems, utilizing symmetric encryption and hash functions. While effective for two to three parties or IoT devices, their scheme encounters performance issues as the number of sensor nodes increases. AKA processes become computationally intensive, leading to higher computation costs. In [14], a fuzzy extractor-based AKA mechanism is presented, but it suffers from vulnerability to password-guessing attacks. The authors in [15] devised a system tailored for healthcare systems. They implemented lattice-based cryptography to fortify the scheme against potential quantum computation threats. Nevertheless, upon examination, it is pinpointed that their approach is vulnerable to impersonation, de-synchronization, and smart-card theft attacks. The authors in [16] have developed an authentication scheme for

Table 1
Summary of Various Related AKA Mechanisms Designed for Healthcare Application.

AKA Mechanism	Limitations of the Existing AKA Mechanisms/Schemes
Ref. [40]	The scheme has a design flaw and weak against the privileged insider attack.
Ref. [41]	The scheme cannot resist the de-synchronization and impersonation attacks.
Ref. [42]	The scheme cannot resist the MITM and impersonation attacks.
Ref. [43]	Unable to thwart impersonation and privileged insider attacks.
Ref. [44]	Unable to resist password-guessing attacks.
Ref. [45]	Unable to thwart DoS and replay attacks.
Ref. [46]	Unable to thwart MITM and session key leakage attacks.
Ref. [14]	Suffers from vulnerability to password-guessing attack
Ref. [47]	Cannot prevent forgery and MITM attack
Ref. [29]	Cannot prevent impersonation, and insider attacks

healthcare systems utilizing post-quantum cryptography. They confirmed the security of their proposed scheme through validation using the Scyther tool and the random oracle model.

The AKA mechanism outlined in [17], [18], [19], and [20], smart cards retain an explicit password validation parameter, rendering them susceptible to offline password-guessing attacks. Conversely, [21] and [22] lack password validation parameters, leaving them vulnerable to denial-of-service attacks. Despite [18] employing only three chaotic operations, it still falls short of ensuring perfect forward security in cases of long-term private key leaks. Additionally, despite utilizing elliptic curve cryptography, [22] stores a key in a storage device, undermining forward secrecy protection. [21] fails to provide anonymity due to plaintext identity transmission on public channels and cannot withstand clock synchronization attacks. Likewise, [17], [18], [19], [21], and [20], are incapable of resisting offline password-guessing attacks, thus failing to provide three-factor security directly. While [22] can resist such attacks, it remains susceptible to key-compromised impersonation and lacks forward secrecy. The AKA mechanism proposed in [23] is prone to key compromise user impersonation and clock synchronization attacks. The AKA protocol in [19] is vulnerable to offline password guessing and key-compromise user impersonation attacks and lacks smart-card revocation capabilities. Similarly, the AKA protocol in [17] is deficient in resisting offline password guessing, session-specific temporary information attacks, and clock synchronization attacks, and lacks both three-factor security and smart-card revocation functions. The authors in [24], proposed an AKA mechanism for the smart healthcare system using symmetric encryption and one way hash function, and its security is corroborated using the Scyther tool and ROR model. In [25], an authentication scheme tailored for low-power mobile devices is introduced. Unfortunately, it is vulnerable to password brute-force attacks due to essential security oversights. In response, the authors in [26], propose a lightweight mutual authentication scheme to establish a secure channel between users and their devices. While this secures network data from unauthorized access, it remains susceptible to device capture attacks. Another authentication approach, pioneered by the authors in [27], leverages biometric data for network node authentication. This method enhances security by integrating the patient's electrocardiogram signals into the authentication process. However, it grapples with issues of untraceability and key escrow. To address these shortcomings, the authors in [28], refine the aforementioned scheme by introducing anonymous AKA technique. Despite these improvements, scalability remains a challenge due to significant communication and computation overheads.

The authors in [29], proposed an AKA mechanism for the smart healthcare system using symmetric encryption and one way hash function, and its security is corroborated using the Scyther tool and ROR model. The authors of [30] introduced a two-factor AKA mechanism for healthcare applications on wireless sensor networks, utilizing symmetric encryption and decryption. However, despite claims of robustness, in [31], the authors identified vulnerabilities to offline password-guessing and privileged insider attacks within the protocol [30]. Furthermore, in [31], it is highlighted the absence of user anonymity in [30]. Subsequently, in [32], the authors argued that the AKA mechanism proposed in [31] is susceptible to offline password-guessing, user impersonation, and sensor node capture attacks. Nevertheless, in [33], the authors pointed out vulnerabilities in [32], including susceptibility to stolen smart card attacks, offline password-guessing attacks, user impersonation attacks, and DoS attacks, alongside ineffective mutual authentication. In [34], the authors introduced a secure patient monitoring system, yet [35] demonstrated vulnerabilities to offline password-guessing, user imitation, and known session-specific temporary information attacks within this protocol. Additionally, the authors in [36] proposed a lightweight AKA mechanism, but [37] identified susceptibility to sensor node capture attacks and inadequate authentication between users and devices. Furthermore, in [38], the authors proposed an AKA mechanism for ambient assisted medical living systems. The authors in [39], proposed an AKA mechanism for the smart healthcare system using symmetric encryption and one way hash function, and its security is corroborated using the Scyther tool and ROR model. A summary of these significant related works is provided in Table 1.

The scheme proposed in [40] has a design flaw and is susceptible to privileged insider attacks. Similarly, the scheme in [41] cannot resist de-synchronization and impersonation attacks. The protocol described in [42] fails to defend against MITM and impersonation attacks. The scheme introduced in [43] is unable to thwart impersonation and privileged insider attacks. Additionally, the technique presented in [44] is vulnerable to password-guessing attacks. In the work in [45], the scheme cannot prevent DoS and replay attacks. Finally, the protocol discussed in [46] is unable to protect against MITM and session key leakage attacks.

2.1. Motivation and research contribution

After investigating various AKA mechanisms as discussed in Section 2, it becomes evident that these mechanisms lack security against multiple attacks, including session key, replay, and impersonation. Moreover, they suffer from key escrow issues. Additionally, they fail to assure perfect forward secrecy and are vulnerable to smart card and privileged insider attacks. Furthermore, they lack verification of protocol using established models or tools like the ROR model, BAN logic, or AVISPA. Drawing from these observations and identified security gaps, we propose two AKA mechanisms to address specific needs within the context of a smart patient monitoring system. The first mechanism focuses on retrieving patient data from IoT devices deployed in the system and securely storing this data on the cloud server. The second mechanism enables users, such as doctors and nurses, to securely retrieve the stored data from the cloud server. The main contributions of the paper include:

- We introduce two mechanisms: IoTD-2-CS (IoT device-2-cloud server) and UX-2-CS (user-2-cloud server). These mechanisms employ symmetric encryption, PUF, and hash functions. IoTD-2-CS ensures the authenticity of IoT devices, establishing session keys between the IoT device and cloud server. Similarly, UX-2-CS verifies the authenticity of users, establishing session keys between the user and cloud server. The session keys established by these mechanisms ensure secure and unintelligible communication between IoT devices and cloud servers, as well as between users and cloud servers. The integration of PUF functionality enhances the physical security of the system.
- Both informal and formal security analyses of the proposed AKA mechanisms are performed to illustrate their resilience against various security attacks. It is demonstrated that both AKA mechanisms exhibit robust security resistance against MITM, replay, and impersonation attacks.
- The proposed UX-2-CS AKA mechanism is compared to closely related AKA mechanisms Ref. [48], Ref. [49], Ref. [50], and Ref. [51] in terms of computational cost, communication cost, and security functionalities. UX-2-CS AKA mechanism demonstrates a reduction in computational cost by 69 to 86.25 percent and a decrease in communication cost by 25.55 to 56.85 percent. Similarly, the AKA mechanisms for IoTD-2-CS servers are compared with Ref. [52], Ref. [53], Ref. [54], and Ref. [55] regarding computational cost, communication cost, and security functionalities. Additionally, the AKA mechanism for IoTD-2-CS shows a decrease in computational cost by 75 to 95.80 percent and a reduction in communication cost by 35.24 to 74.62 percent. UX-2-CS and IoTD-2-CS AKA mechanisms provide enhanced security features.

2.2. Paper outline

The paper is structured as follows: In Section 3, we present the system models and background knowledge required to elaborate on the proposed IoTD-2-CS and UX-2-CS AKA mechanisms. The detailed construction of both IoTD-2-CS and UX-2-CS AKA mechanisms is elaborated in Section 4. The security analysis of IoTD-2-CS and UX-2-CS AKA mechanisms is performed using both formal and informal methods in Section 5. The performance comparison of IoTD-2-CS and UX-2-CS AKA mechanisms is conducted in Section 6. The paper concludes with final remarks in Section 7.

3. System model and background knowledge

3.1. Network model

The presented AKA mechanism's network model, as depicted in Fig. 2, incorporates key components: user U_x, IoT devices ITD_y, and cloud server CS_y. Moreover, a trusted authority oversees the registration of U_x, ITD_y, and CS_y before their deployment in the smart healthcare application.

ITD_y is tasked with collecting sensitive patient data within the hospital and transmitting it to CS_y. Typically deployed in smart patient monitoring systems or worn by patients, these devices gather data about the patient and deliver it to CS_y through an public wireless or wired communication link.

U_x includes nurses/doctors, home users, and policymakers who need to access data stored on CS_y to make informed decisions. Communication between U_x and CS_y occurs via the public wireless or wired communication link. Thus it is imperative to ensure the integrity of retrieved information by U_x from CS_y and information stored by ITD_y on CS_y.

Both U_x and ITD_y exchange information with CS_y via the public wireless or wired communication link, which is vulnerable to various security threats. Attackers could intercept data transmitted over these channels, leading to potential security breaches. To mitigate this risk and prevent unauthorized access to communicated data, an AKA mechanism is imperative. For the smooth reading of the paper a list of notation is provided in Table 2.

3.2. Adversarial model

The widely recognized "Dolev-Yao threat (DY) model" [56,57] enables an attacker, designated as $\mathcal{A}$, to intercept communicated messages and manipulate them by modifying, deleting, or inserting fabricated data when communicating with other parties such as a patient (patient), doctor, and cloud server. Additionally, the CK-adversary model [10] is also considered as a paramount threat model, more contemporary in comparison to the DY model. In the CK-adversary model, attacker $\mathcal{A}$ may compromise the confidential credentials shared among interacting participants, potentially leading to session hijacking attacks and the compromise of session

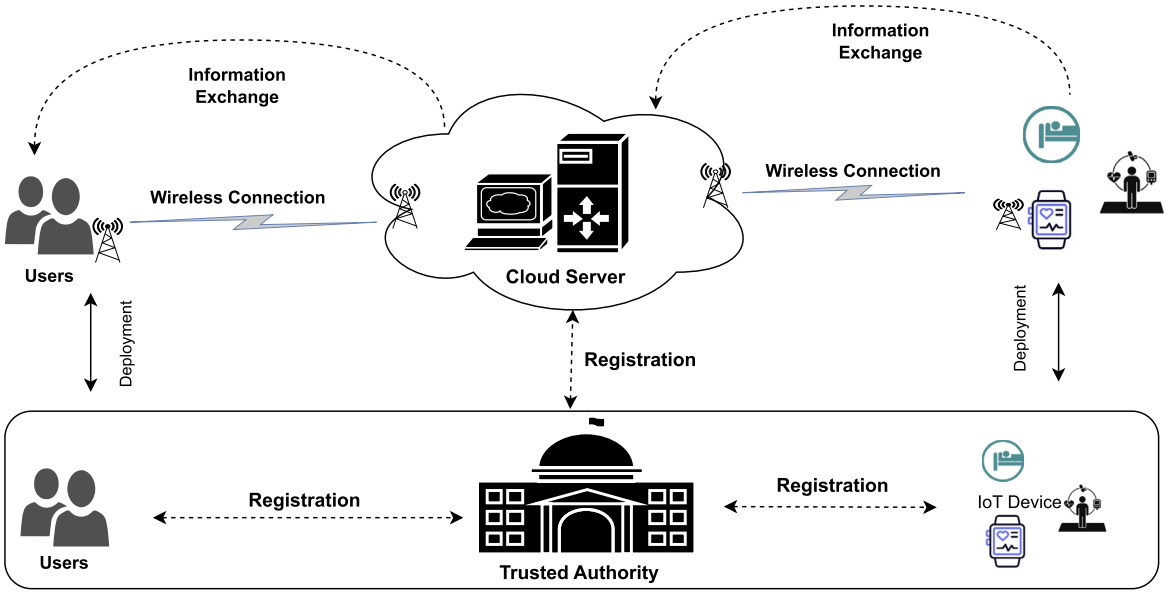


Fig. 2. Network model for user and IoT device authentication.

Table 2
Notations.

Notation	Description
U_x	User
RA	Registration authority
CS_Y	Cloud server
PSK_{CS_y}	Long-term secret key of the CS_y
PW_{U_x}	password of user
$\oplus$	XOR function
CH_{U_x}	Challenge
R_{U_x}	Response
CH_{CS_y}	Challenge
R_{CS_y}	Response generated at CS_y
$PUF(\cdot)$	PUF function
$E_k(data)$	Encryption of data with secret key K
$D_k(ct)$	Decryption of ciphertext ct with secret key K
Y_1, Y_2, Y_6 *to* Y_9, Y_{12} *to* Y_{14}	Cipher texts generated by encryption algorithm in IoTD-2-CS AKA mechanism
W_1, W_2, W_8 *to* W_{11}, W_{14} *to* W_{17}	Cipher texts generated by encryption algorithm in UX-2-CS AKA mechanism
$\mathcal{A}$	Adversary or attacker
ID_i	Identity of the IoT device
Bio_{U_x}	Biometric information of U_i
$\|\|$	Concatenation operation
RGD_{CS_y}	Regeneration data for the biometric key at CS_y
RGD_{U_x}	Regeneration data for the biometric key at U_x
$Gen(\cdot)$	FE based biometric key generation algorithm
$Rep(\cdot)$	FE based biometric key regeneration algorithm
σ_{U_x}	Biometric key of the user
IV	Initialization vector
$H(\cdot)$	Hash function

states and keys. Hence, when designing the proposed AKA mechanism, we ensured it offers both forward and backward secrecy, even in scenarios where $\mathcal{A}$ can compromise the current session key.

There is a possibility that a user's smart card or mobile device could be lost or stolen by $\mathcal{A}$, enabling the extraction of all stored credentials through power analysis attacks. It is assumed that $\mathcal{A}$ can only guess either a low-entropy password or the identity of a patient at a given time, but not both simultaneously. Furthermore, it is anticipated that predicting the personal biometric and its corresponding biometric secret key would be significantly more challenging compared to guessing a low-entropy password of a patient [58,59].

3.2.1. Security requirements

Here are the security requirements to consider when designing the IoTD-2-CS and UX-2-CS AKA mechanisms.

- Mutual Authentication: The authentication mechanism is a necessary security measure in any system, guaranteeing that only authorized users or devices can access resources or services. In the context of IoTD-2-CS and UX-2-CS AKA mechanisms, robust authentication mechanisms are crucial to ascertain the identity of both IoT devices and users. This assists prevent unauthorized access, data breaches, and other security risks.
- Session Key Establishment: Throughout the authentication process, network entities establish a session key to facilitate encrypted communication for subsequent interactions. In the IoTD-2-CS and UX-2-CS AKA mechanism, IoT devices establish a session key with the cloud server to transmit patient data in an encrypted format. Authorized users, such as doctors and nurses, can securely access this encrypted data from the cloud server via the public Internet.
- Resistance to Various Security Attacks: The IoTD-2-CS and UX-2-CS AKA mechanisms need to withstand a range of security threats, including MITM attacks, replay attacks, impersonation attempts, DoS attacks, and insider attacks by privileged users.

3.3. Background knowledge

In this subsection, we will elaborate on the various cryptographic primitives used in designing the AKA mechanisms.

3.3.1. PUF

Physically unclonable functions (PUFs) are admiringly practical for hardware security due to their capability to develop unique hardware fingerprints. The inherent unpredictability caused by process variations during IC manufacturing makes PUFs impossible to replicate. Additionally, PUFs provide robust protection against physical tampering. They are widely used in applications such as random number generation, secret-key generation, and device authentication. PUFs are characterized by their uniqueness and reliability: uniqueness ensures that identical PUF functions on different devices produce distinct outputs, while reliability guarantees consistent responses to the same challenge over time [60–62].

3.3.2. FE

A fuzzy extractor is a cryptographic primitive designed to generate stable and secure keys from noisy or imprecise data, such as biometric information or physical unclonable functions (PUFs). This concept is crucial for enhancing security in systems where the exact reproducibility of input data is challenging.

Noisy Data Handling: Traditional cryptographic key generation requires precise input data, which is impractical for bio metrics and other noisy sources. Fuzzy extractors address this issue by reliably generating the same cryptographic key even from slightly different versions of the input data.

Error Tolerance: Fuzzy extractors can tolerate a certain amount of error in the input data, making them ideal for use with bio-metrics, which can vary slightly with each measurement due to factors like environmental conditions or user interaction [61,62].

The following are the main functions of the FE: Generation ($Gen(\cdot)$): Takes a noisy input and produces a stable key and a helper string. The helper string does not reveal the key but is used to recover the key from similar inputs. Reproduction ($Rep(\cdot)$): Uses the noisy input and the helper string to reproduce the original key. This process ensures that the same key is generated from inputs that are close to the original.

4. The proposed AKA mechanisms

This section introduces two AKA mechanisms. The first one is the UX-2-CS AKA mechanism, which handles the generation of the session key between the user and the cloud server following the validation of the user's authenticity. The second mechanism is responsible for generating the session key between the IoT device and the cloud server after verifying the authenticity of the IoT device. Both AKA mechanisms are further explained in the following subsections.

4.1. Cloud server registration

The trusted authority selects a unique long-term secret key PSK_{cs_y} for CS_y. Additionally, the trusted authority selects CH_{cs_y} and sends PSK_{cs_y} and CH_{cs_y} to CS_y securely. Moreover, CS_y computes $R_{cs_y} = PUF(CH_{cs_y})$, $(K_{cs_y}, RGD_{cs_y}) = Gen(R_{cs_y})$ and $K_y = H(K_{cs_y} \parallel PSK_{cs_y})$. Finally, CS_y keeps $\{RGD_{cs_y}, CH_{cs_y}, PSK_{cs_y}\}$ in its own database.

4.2. U_x registration

The trusted authority is responsible for registering the user U_x before its deployment in the smart healthcare applications. U_x selects its identity ID_{U_x}, password PW_{U_x}, and imprints its own biometric information Bio_{U_x} on the smart device owned by U_x. After getting ID_{U_x}, PW_{U_x}, and Bio_{U_x}, U_x selects challenge CH_{U_x} and computes response $R_{U_x} = PUF(CH_{U_x})$, biometric key $(\sigma_{U_x}, RGD_{U_x}) = Gen(Bio_{U_x})$, $K_2 = H(PW_{U_x} \parallel \sigma_{U_x})$, and sends K_2, CH_{U_x}, and R_{U_x} to the CS_y securely. CS_y after getting K_2 selects random number R_{ux} and computes $(W_1, W_2) = E_{K_y}\{(IV_{cs_y} = K_y), (K_2 \oplus R_{ux}, R_{ux})\}$ and sends (W_1, W_2) to the U_x securely. In addition CS_y computes $PID_x = H(K_2)$ and stores the parameters $\{PID_x, CH_{U_x}, R_{U_x}\}$ in its own database. Moreover, U_x computes $IV_2 = H(PW_{U_x} \parallel ID_{U_x})$, $(Ctx_2) = E_{K_2}\{IV_2, W_1, W_2\}$, and $W_4 = H(W_1 \parallel W_2 \parallel K_2 \parallel IV_2)$. Finally, U_x stores the credentials $\{Ctx_2, (W_1, W_2), RGD_{U_x}, W_4, Gen(\cdot), Rep(\cdot)\}$ it its device's memory.

4.3. ITD_k registration

The trusted authority is responsible for registering the IoT device before its deployment in the smart healthcare application. It selects a long-term secret key K_d, identity ID_i, and challenge CH_i, and sends ID_i, K_d, and CH_i to ITD_k. Additionally, ITD_k generates the response $R_p = PUF(CH_i)$ and securely transmits ID_i, CH_i, and R_p back to the trusted authority. The trusted authority then computes $(Y_1, Y_2) = E_{K_y}\{(IV_j = K_y), (ID_i \oplus R_{x1}, R_{x1})\}$, where R_{x1} is a random number. Additionally, the trusted authority computes $SID_i = H(ID_i)$ and stores the parameters $\{SID_i, CH_i, R_p\}$ in the database of CS_y. Moreover, the trusted authority securely transmits $\{Y_1, Y_2, K_d, ID_i\}$ to ITD_k.

4.4. IoTD-2-CS AKA mechanism

In this phase, the session key between the IoT device and the cloud server is generated following mutual authentication. IoT devices can securely store information on the cloud server using the established session key. The steps outlined below are crucial for both session key generation and mutual authentication.

4.4.1. Step-1

When the IoT device has data to transmit to CS_y, it initiates the mechanism known as IoTD-2-CS. In this process, the IoT device selects the timestamp T_x and computes.

$$Y_4 = H(Y_1 \parallel Y_2 \parallel ID_i \parallel T_x). \tag{1}$$

It is important to emphasize that Y_4 serves as the authentication parameter, verifying the integrity of the message at CS_y. Furthermore, ITD_k generates a message M_1 containing $\{T_x, Y_1, Y_2, Y_4\}$ and transmits it to CS_y through the public wireless or wired communication link.

4.4.2. Step-2

Upon receiving the message M_1 from the IoT device, CS_y verifies the freshness of M_1 using the condition $T_{dl} \geq |T_{re} - T_x|$. If this condition is met, CS_y proceeds to compute the following:

$$T_{dl} \geq |T_{re} - T_x| \tag{2}$$

$$R^*_{cs_y} = PUF(CH_{cs_y}), \tag{3}$$

$$K^*_{cs_y} = Rep(K^*_{cs_y}, RGD_{cs_y}), \tag{4}$$

$$K_y = H(K^*_{cs_y} \parallel PSK_{cs_y}), \tag{5}$$

$$(ID_i \oplus R_x, R_x) = D_{K_y}\{(IV_i = K_y), Y_1, Y_2\}, \tag{6}$$

$$Y_4 = H(Y_1 \parallel Y_2 \parallel ID_i \parallel T_x), \tag{7}$$

$$Y_5 \stackrel{?}{=} Y_4. \tag{8}$$

It is worth mentioning that in the proposed IoTD-2-CS AKA mechanism we employed the symmetric encryption and decryption algorithm referred to as AES-CBC. Prior to authenticating the received message M_1, CS_y must calculate its secret key K_y, preceded by the computation of $R^*_{cs_y}$ and $K^*_{cs_y}$. Subsequently, after the decryption operation, CS_y obtains the ID_i of the IoT device and computes the authentication parameter Y_4. Additionally, the integrity of the message M_1 is verified through condition (8). Moreover, CS_y computes $SID_i = H(ID_i)$ and retrieves $\{CH_i, R_p\}$ from its own database. CS_y creates the response message M_2 by selecting the values R_{x1}, R_m, and timestamps T_y through the following computations:

$$Y^n_1 = (ID_i \oplus R^n_{x1}), \tag{9}$$

$$(Y^n_1, Y^n_2) = E_{K_y}\{(IV_j = K_y), (ID_i \oplus R^n_{x1}, R^n_{x1})\}, \tag{10}$$

$$(Y_6, Y_7, Y_8, Y_9) = E_{K_d}\{(IV_k = Y_1), Y^n_1, Y^n_2, R_m, R_p\}, \tag{11}$$

$$Y_{10} = H(Y_1^n \parallel Y_2^n \parallel R_m \parallel CH_i \parallel T_y \parallel K_d). \tag{12}$$

Furthermore, CS_y generates a message M_2 containing $\{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}$ and transmits it to ITD_k through the public wireless or wired communication link.

4.4.3. Step-3

Upon reception of the message M_2 from the IoT device, CS_y validates the freshness of M_2 by applying the condition $T_{dl} \geq |T_{re} - T_y|$. If this condition holds true, CS_y then proceeds to perform the following computations:

$$(Y_1^n, Y_2^n, R_m, R_p) = D_{K_d}\{(IV_l = Y_1), Y_6, Y_7, Y_8, Y_9\}, \tag{13}$$

$$Y_{11} = H(Y_1^n \parallel Y_2^n \parallel R_m \parallel R_p \parallel T_y \parallel K_d), \tag{14}$$

$$Y_{11} \stackrel{?}{=} Y_{10}. \tag{15}$$

The authenticity of the received message M_2 is confirmed by checking condition (15). If this condition is satisfied, CS_y proceeds to perform the following computations after selecting T_z, R_n, and CH_i^n.

$$(K_{d1}, RGD_i^*) = Gen(R_p), \tag{16}$$

$$R_i^n = PUF(CH_i^n), \tag{17}$$

$$(Y_{12}, Y_{13}, Y_{14}) = E_{K_{d1}}\{(IV_m = RGD_i^*), R_p^n, CH_i^n, R_n\}, \tag{18}$$

$$SK_i = H(K_{d1} \parallel CH_i^n \parallel R_p^n \parallel R_n \parallel R_m), \tag{19}$$

$$Y_{15} = H(RGD_i^* \parallel R_p^n \parallel CH_i^n \parallel R_n \parallel K_{d1} \parallel SK_i). \tag{20}$$

After completing the aforementioned computations, ITD_k constructs the message M_3 with $\{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}$ and sends it to CS_y via the public wireless or wired communication link.

4.4.4. Step-3

Upon reception of the message M_2 from the IoT device, CS_y validates the freshness of M_2 by applying the condition $T_{dl} \geq |T_{re} - T_z|$. If this condition holds true, CS_y then proceeds to perform the following computations:

$$(K_{d1}) = Rep(R_p, RGD_i^*), \tag{21}$$

$$(R_p^n, CH_i^n, R_n) = E_{K_{d1}}\{(IV_n = RGD_i^*), Y_{12}, Y_{13}, Y_{14}\}, \tag{22}$$

$$SK_y = H(K_{d1} \parallel CH_i^n \parallel R_p^n \parallel R_n \parallel R_m), \tag{23}$$

$$Y_{16} = H(RGD_i^* \parallel R_p^n \parallel CH_i^n \parallel R_n \parallel K_{d1} \parallel SK_y), \tag{24}$$

$$Y_{15} \stackrel{?}{=} Y_{16}. \tag{25}$$

Both CS_y and ITD_k compute the session keys SK_i and SK_y for encrypted communication in the future. The authenticity of the message M_3 is verified using condition (25). The fulfillment of this condition also signifies successful authentication. Finally, CS_y replaces (R_i^n, CH_i^n) with (R_i, CH_i) in its own database. The summary of the IoTD-2-CS is given in Fig. 3.

4.5. UX-2-CS AKA mechanism

During this phase, the generation of the session key between the user and the cloud server occurs after mutual authentication has been achieved. Users such as doctors, nurses, and others can securely access information stored on the cloud server. The following steps are essential for both session key generation and mutual authentication.

4.5.1. Step-1

The user U_x starts the AKA mechanism by taking Bio_{U_x}, RGD_{U_x}, ID_{U_x}, and PW_{U_x} as the input parameters. U_x computes the following:

$$(\sigma_{U_x}) = Rep(Bio_{U_x}, RGD_{U_x}), \tag{26}$$

$$K_2 = H(PW_{U_x} \parallel \sigma_{U_x}), \tag{27}$$

$$IV_2 = H(PW_{U_x} \parallel ID_{U_x}), \tag{28}$$

$$(W_1, W_2) = D_{K_2}\{IV_2, Ctx_2\}, \tag{29}$$

$$W_4 = H(W_1 \parallel W_2 \parallel K_2 \parallel IV_2), \tag{30}$$

$$W_4 \stackrel{?}{=} W_2. \tag{31}$$

IoT device ITD_k	**Medical Cloud Server**/CS_y
$\{Y_1, Y_2, K_d, ID_i\}$	$\{\{PSK_{cs_y}, PID_x, (CH_{CS_y}, RGD_{CS_y}), (SID_i, CH_i, R_p)\}\}$
selects T_x, computes $Y_4 = H(Y_1 \| Y_2 \| ID_i \| T_x)$, $\xrightarrow[(ITD_k \rightarrow CS_y)]{M_1:\{T_x, Y_1, Y_2, Y_4\}}$.	checks $T_{dl} \geq \lvert T_{re} - T_x \rvert$, if holds, computes $R^*_{cs_y} = PUF(CH_{cs_y})$, computes $K^*_{cs_y} = Rep(K^*_{cs_y}, RGD_{cs_y})$, computes $K_y = H(K^*_{cs_y} \| PSK_{cs_y})$, computes $(ID_i \oplus R_x, R_x) = D_{K_y}\{(IV_i = K_y), Y_1, Y_2\}$, computes $Y_4 = H(Y_1 \| Y_2 \| ID_i \| T_x)$, checks $Y_5 \stackrel{?}{=} Y_4$, if holds, retrieves SID_i from the database, selects R_{x1}, R_m, and timestamps T_y, computes$Y_1^n = (ID_i \oplus R_{x1}^n)$, computes $(Y_1^n, Y_2^n) = E_{K_y}\{(IV_j = K_y), (ID_i \oplus R_{x1}^n, R_{x1}^n)\}$, computes $(Y_6, Y_7, Y_8, Y_9) = E_{K_d}\{(IV_k = Y_1), Y_1^n, Y_2^n, R_m, R_p\}$, computes $Y_{10} = H(Y_1^n \| Y_2^n \| R_m \| CH_i \| T_y \| K_d)$.
checks $T_{dl} \geq \lvert T_{re} - T_y \rvert$, If valid, computes $(Y_1^n, Y_2^n, R_m, R_p) = D_{K_d}\{(IV_l = Y_1), Y_6, Y_7, Y_8, Y_9\}$, computes $Y_{11} = H(Y_1^n \| Y_2^n \| R_m \| R_p \| T_y \| K_d)$, checks $Y_{11} \stackrel{?}{=} Y_{10}$, computes $(K_{d1}, RGD_i^*) = Gen(R_p)$, computes $R_i^n = PUF(CH_i^n)$, computes $(Y_{12}, Y_{13}, Y_{14}) = E_{K_{d1}}\{(IV_m = RGD_i^*), R_p^n, CH_i^n, R_n\}$, computes $SK_i = H(K_{d1} \| CH_i^n \| R_p^n \| R_n \| R_m)$, computes $Y_{15} = H(RGD_i^* \| R_p^n \| CH_i^n \| R_n \| K_{d1} \| SK_i)$, $\xrightarrow[(ITD_k \rightarrow CS_y)]{M_3:\{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}}$.	$\xleftarrow[(CS_y \leftarrow ITD_k)]{M_2:\{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}}$. checks $T_{dl} \geq \lvert T_{re} - T_z \rvert$, if holds, computes $(K_{d1}) = Rep(R_p, RGD_i^*)$, computes $(R_p^n, CH_i^n, R_n) = E_{K_{d1}}\{(IV_n = RGD_i^*), Y_{12}, Y_{13}, Y_{14}\}$, computes $SK_y = H(K_{d1} \| CH_i^n \| R_p^n \| R_n \| R_m)$, computes $Y_{16} = H(RGD_i^* \| R_p^n \| CH_i^n \| R_n \| K_{d1} \| SK_y)$, computes $Y_{15} \stackrel{?}{=} Y_{16}$, updates (R_i^n, CH_i^n) with (R_i, CH_i).
$SK_i (= SK_y) = H(K_{d1} \| CH_i^n \| R_p^n \| R_n \| R_m)$	

Fig. 3. AKA phase of IoTD-2-CS.

It is vital to state that σ_{U_x} serves as the biometrically derived secret key, generated through the reproduction function of FE. In the decryption algorithm, K_2 represents the secret key. IV_2 stands for the initialization vector. Additionally, W_4 functions as the authentication parameter, its validity assessed through the condition (31). Fulfillment of this condition indicates successful local verification of the user. Furthermore, upon meeting this criterion, U_x proceeds to select T_g, R_i, and calculates the following:

$$W_5 = H(K_2 \| R_i) \oplus H(K_2 \| T_g), \tag{32}$$

$$W_6 = H(H(K_2 \| R_i) \| T_g \| W_1 \| W_2). \tag{33}$$

The above computed W_5 and W_6 are the component of the AKA message sent by U_x to CS_y. U_x generates the message $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$ and sends this generated message to CS_y via the public wireless or wired communication link.

4.5.2. Step-2

CS_y obtains the message $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$ and extracts the timestamp. It then ascertains its freshness by comparing it with the condition $T_{dl} \geq |T_{re} - T_g|$. If this condition is met, the CS_y proceeds to compute the following:

$$R^*_{cs_y} = PUF(CH_{cs_y}), \tag{34}$$

$$K^*_{cs_y} = Rep(R^*_{cs_y}, RGD_{cs_y}), \tag{35}$$

$$K_y = H(K^*_{cs_y} \| PSK_{cs_y}), \tag{36}$$

$$IV_{cs_y} = H(ID_{cs_y} \| K_y), \tag{37}$$

$$(K_2, R_{ux}) = D_{K_y}\{(IV_{cs_y}), W_1, W_2\}, \tag{38}$$

$$H(K_2 \| R_i) = W_5 \oplus H(K_2 \| T_g), \tag{39}$$

$$W_7 = H(H(K_2 \| R_i) \| T_g \| W_1 \| W_2), \tag{40}$$

$$W_6 \stackrel{?}{=} W_7. \tag{41}$$

In the preceding calculations, Rcs_y denotes the response generated by the PUF function, utilizing $CHcs_y$ as its input, assigned to CS_y during registration. The stable key $K^*_{cs_y}$ is reproduced using the FE reproduction function, with $R^*_{cs_y}$ as the input parameter. Additionally, CS_y computes the decryption key K_y and the initialization vector IV_{cs_y}. CS_y gets K_2, R_{ux} and $H(K_2 \| R_i)$ from the decryption process. Furthermore, CS_y calculates the authentication parameter W_7, and the message validity is verified through

condition (41). If this condition holds true, CS_y proceeds to compute $PID_x = H(K_2)$, checks its existence in the database, and if found, retrieves (CH_{U_x}, R_{U_x}). Moreover, CS_y selects T_h, R_j, and R_{ux}^n and computes the followings:

$$K_e = H(H(K_2 \parallel R_i) \parallel K_2) \tag{42}$$

$$(W_1^n, W_2^n) = E_{K_y}\{(IV_{cs_y}), (K_2 \oplus R_{ux}^n), R_{ux}^n\} \tag{43}$$

$$(W_8, W_9, W_{10}, W_{11}) = E_{K_e}\{IV_2 = K_2, CH_{U_x}, R_j, W_1^n, W_2^n\} \tag{44}$$

$$W_{12} = H(R_j \parallel CH_{U_x} \parallel H(K_2 \parallel R_i) \parallel W_8 \parallel W_9 \parallel W_1^n \parallel W_2^n), \tag{45}$$

Here, the encryption key K_e is computed, and new parameters W_1^n and W_2^n are generated using this secret encryption key. Additionally, using the secret encryption key K_y, the parameters W_8 and W_9 are generated through the encryption process. Furthermore, the authentication parameter W_{10} is computed using the hash function. Finally, CS_y constructs a message with parameters Mg_2: $\{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$ and transmit it to U_x.

4.5.3. Step-3

U_x verifies the freshness of the received message based on the condition $T_{dl} \geq |T_{re} - T_h|$. If true, U_x computes the following:

$$K_f = H(H(K_2 \parallel R_i) \parallel K_2), \tag{46}$$

$$(CH_{U_x}, R_j, W_1^n, W_2^n) = D_{K_f}\{(IV_3 = K_2), W_8, W_9, W_{10}, W_{11}\} \tag{47}$$

$$W_{13} = H(R_j \parallel CH_{U_x} \parallel H(K_2 \parallel R_i) \parallel W_8 \parallel W_9 \parallel W_1^n \parallel W_2^n) \tag{48}$$

$$W_{12} \stackrel{?}{=} W_{13},. \tag{49}$$

When K_f serves as the decryption key, the decryption process yields CH_{U_x} and R_j. Furthermore, it computes W_{11}, representing the authentication parameter, while verifying the authenticity and integrity of message Mg_2 through condition (49). If the criterion is fulfilled, and U_x chooses R_k, $CH_{U_x}^n$, and T_i. U_x computes the following:

$$(K_z, RGD_{U_x}^*) = Gen(CH_{U_x}), \tag{50}$$

$$R_{U_x}^n = PUF(CH_{U_x}^n), \tag{51}$$

$$(W_{12}, W_{13}, W_{14}, W_{15}) = E_{K_f}\{(IV_3 = K_2), RGD_{U_x}^*, R_{U_x}^n, CH_{U_x}^n, R_k\}, \tag{52}$$

$$SK_{U_x} = H(K_f \parallel CH_{U_x}^n \parallel R_{U_x}^n \parallel K_2 \parallel R_k \parallel R_i \parallel R_j), \tag{53}$$

$$W_{16} = H(RGD_{U_x}^* \parallel R_{U_x}^n \parallel CH_{U_x}^n \parallel R_k \parallel K_z \parallel SK_{U_x}), \tag{54}$$

$$(Ctx_3) = E_{K_2}\{IV_2, W_1^n, W_2^n\}, \tag{55}$$

$$W_4^n = H(W_1^n \parallel W_2^n \parallel K_2 \parallel IV_2). \tag{56}$$

Through the FE generation function, parameters K_z and $RGD_{U_x}^*$ are derived. Additionally, U_x computes $R_{U_x}^n$ using the PUF function. Subsequently, W_{12}, W_{13}, W_{14}, and W_{15} are generated as cipher-texts using the encryption algorithm, with $IV_3 = K_2$ serving as the initialization vector. A session key is then computed to facilitate encrypted communication, which occurs following mutual authentication. Finally, U_x obtains the authentication credential W_{16} and assembles the message $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$ and deliver it to CS_y. In addition, U_x updates Ctx_3 with Ctx_2 and W_4^n with W_4,

4.5.4. Step-4

After getting Mg_3, CS_y verifies the freshness of the message based on the condition $T_{dl} \geq |T_{re} - T_i|$. If the message is fresh then CS_y computes the following:

$$(RGD_{U_x}^*, R_{U_x}^n, CH_{U_x}^n, R_k) = D_{K_f}\{(IV_5 = K_2), W_{12}, W_{13}, W_{14}, W_{15}\}, \tag{57}$$

$$K_z^* = Rep(CH_{U_x}, RGD_{U_x}^*), \tag{58}$$

$$SK_{CS_y} = H(K_f \parallel CH_{U_x}^n \parallel R_{U_x}^n \parallel K_2 \parallel R_k \parallel R_i \parallel R_j), \tag{59}$$

$$W_{17} = H(RGD_{U_x}^* \parallel R_{U_x}^n \parallel CH_{U_x}^n \parallel R_k \parallel K_z^* \parallel SK_{CS_y}), \tag{60}$$

$$W_{16} \stackrel{?}{=} W_{17}. \tag{61}$$

During the decryption process, CS_y obtains $(RGD_{U_x}, R_{U_x}^n, CH_{U_x}^n, R_k)$ and derives K_z^* using the reproduction function of FE. Subsequently, a session key is computed to enable encrypted communication between U_x and CS_y post mutual authentication. Following the computation of the authentication parameter W_{17}, CS_y verifies the integrity of the received message. If the message is authenticated, CS_y updates its own database with $(CH_{U_x}^n, R_{U_x}^n)$, replacing (CH_{U_x}, R_{U_x}). The user AKA mechanism is summarized in Fig. 4.

User U_x	**Medical Cloud Server**/CS_y
$\{Ctx_2, RGD_{U_x}, W_4, Gen(\cdot), Rep(\cdot)\}$	$\{PSK_{cs_y}, PID_x, (CH_{CS_y}, RGD_{CS_y}), (CH_{U_x}, R_{U_x}), (PID_x, CH_{U_x}, R_{U_x})\}$
computes U_x enters ID_{U_x}, PW_{U_x}, and imprints Bio_{U_x}, computes $(\sigma_{U_x}) = Rep(Bio_{U_x}, RGD_{U_x})$, computes $K_2 = H(PW_{U_x} \| \sigma_{U_x})$, computes $IV_2 = H(PW_{U_x} \| ID_{U_x})$, computes $(W_1, W_2) = D_{K_2}\{IV_2, Ctx_2\}$, computes $W_4 = H(W_1 \| W_2 \| K_2 \| IV_2)$, checks $W_4 \stackrel{?}{=} W_3$, if holds, selects T_g, R_i, computes $W_5 = H(K_2 \| R_i) \oplus H(K_2 \| T_g)$, computes $W_6 = H(H(K_2 \| R_i) \| T_g \| W_1 \| W_2)$, $\xrightarrow{Mg1:\{T_g, W_1, W_2, W_5, W_6\}}$ $(U_x \rightarrow CS_y)$.	checks $T_{dl} \geq \lvert T_{re} - T_g \rvert$, if satisfied, computes $R^*_{cs_y} = PUF(CH_{cs_y})$, computes $K^*_{cs_y} = Rep(K^*_{cs_y}, RGD_{cs_y})$, computes $K_y = H(K^*_{cs_y} \| PSK_{cs_y})$, computes $IV_{cs_y} = H(ID_{cs_y} \| K_y)$, computes $(K_2, R_{ux}) = D_{K_y}\{(IV_{cs_y}), W_1, W_2\}$, computes $H(K_2 \| R_i) = W_5 \oplus H(K_2 \| T_g)$, computes $W_7 = H(H(K_2 \| R_i) \| T_g \| W_1 \| W_2)$, checks $W_6 \stackrel{?}{=} W_7$, if satisfied, computes $PID_x = H(K_2)$, checks if PID_x exist in database, if found, retrieves (CH_{U_x}, R_{U_x}), selects T_h and R_j, R^n_{ux} computes $K_e = H(H(K_2 \| R_i) \| K_2)$, computes $(W^n_1, W^n_2) = E_{K_y}\{(IV_{cs_y}), (K_2 \oplus R^n_{ux}), R^n_{ux}\}$, computes $(W_8, W_9, W_{10}, W_{11}) = E_{K_e}\{IV_2 = K_2, CH_{U_x}, R_j, W^n_1, W^n_2\}$, computes $W_{12} = H(R_j \| R_{U_x} \| H(K_2 \| R_i) \| W_8 \| W_9 \| W^n_1 \| W^n_2)$, $\xleftarrow{Mg2:\{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}}$ $(U_x \leftarrow CS_y)$.
checks $T_{dl} \geq \lvert T_{re} - T_h \rvert$, If valid, computes $K_f = H(H(K_2 \| R_i) \| K_2)$, computes $(CH_{U_x}, R_j, W^n_1, W^n_2) = D_{K_f}\{(IV_3 = K_2), W_8, W_9, W_{10}, W_{11}\}$, computes $W_{13} = H(R_j \| CH_{U_x} \| H(K_2 \| R_i) \| W_8 \| W_9 \| W^n_1 \| W^n_2)$, checks $W_{12} \stackrel{?}{=} W_{13}$, if satisfied and selects R_k, $CH^n_{U_x}$, and T_i, computes $(K_z, RGD^*_{U_z}) = Gen(R_{U_x})$, $R^n_{U_x} = PUF(CH^n_{U_x})$, computes $(W_{14}, W_{15}, W_{16}, W_{17}) = E_{K_f}\{(IV_4 = K_2), RGD^*_{U_z}, R^n_{U_x}, CH^n_{U_x}, R_k\}$, computes $SK_{U_z} = H(K_f \| CH^n_{U_z} \| R^n_{U_z} \| K_2 \| R_k \| R_i \| R_j)$, computes $W_{18} = H(RGD^*_{U_x} \| R^n_{U_x} \| CH^n_{U_x} \| R_k \| K_z \| SK_{U_x})$, computes $(Ctx_3) = E_{K_2}\{IV_2, W^n_1, W^n_2\}$, computes $W^n_4 = H(W^n_1 \| W^n_2 \| K_2 \| IV_2)$, updates Ctx_3 with Ctx_2 and W^n_4 with W_4, $\xrightarrow{Mg3:\{T_i, W_{14}, W_{15}, W_{16}, W_{17}, W_{18}\}}$ $(U_x \rightarrow CS_y)$.	checks $T_{dl} \geq \lvert T_{re} - T_i \rvert$, If valid, computes $(RGD^*_{U_z}, R^n_{U_x}, CH^n_{U_z}, R_k) = D_{K_f}\{(IV_5 = K_2), W_{14}, W_{15}, W_{16}, W_{17}\}$, computes $K^*_z = Rep(CH_{U_x}, RGD^*_{U_x})$, computes $SK_{CS_y} = H(K_f \| CH^n_{U_z} \| R^n_{U_z} \| K_2 \| R_k \| R_i \| R_j)$, computes $W_{19} = H(RGD^*_{U_x} \| R^n_{U_x} \| CH^n_{U_x} \| R_k \| K^*_z \| SK_{CS_y})$, checks $W_{18} \stackrel{?}{=} W_{19}$, if satisfied, updates (CH_{U_x}, R_{U_x}) with $(CH^n_{U_x}, R^n_{U_x})$.
$SK_{U_x} (= SK_{CS_y}) = H(K_f \| CH^n_{U_x} \| R^n_{U_x} \| K_2 \| R_k \| R_i \| R_j)$	

Fig. 4. AKA phase UX-2-CS.

4.6. Password update mechanism

The proposed UX-2-CS AKA mechanism facilitates password changes and biometric updates. To achieve this, the user U_x utilizes the old $Bio^o U_x$, $RGD^o_{U_x}$, $ID^o_{U_x}$, and $PW^o_{U_x}$ as input parameters, and computes the following:

$$(\sigma^o_{U_x}) = Rep(Bio^o_{U_x}, RGD^o_{U_x}), \tag{62}$$

$$K_2 = H(PW^o_{U_x} \| \sigma^o_{U_x}), \tag{63}$$

$$IV_2 = H(PW^o_{U_x} \| ID^o_{U_x}), \tag{64}$$

$$(W^o_1, W^o_2) = D_{K^o_2}\{IV^o_2, Ctx^o_2\}, \tag{65}$$

$$W^o_4 = H(W^o_1 \| W^o_2 \| K^o_2 \| IV^o_2), \tag{66}$$

$$W^o_4 \stackrel{?}{=} W_2. \tag{67}$$

If condition (67) holds, then U_x generates new and updated secret parameters $Bio^n_{U_x}$, $RGD^n_{U_x}$, $ID^n_{U_x}$, and $PW^n_{U_x}$. U_x computes the following:

$$(\sigma^n_{U_x}, RGD^n_{U_x}) = Gen(Bio^n_{U_x}), \tag{68}$$

$$K^n_2 = H(PW^n_{U_x} \| \sigma^n_{U_x}), \tag{69}$$

$$IV^n_2 = H(PW^n_{U_x} \| ID^n_{U_x}), \tag{70}$$

$$(Ctx^n_2) = E_{K^n_2}\{IV^n_2, W_1, W_2\}, \tag{71}$$

$$W^n_4 = H(W_1 \| W_2 \| K^n_2 \| IV^n_2). \tag{72}$$

Finally, U_x updates with $\{Ctx_2, RGD_{U_x}, W_4, Gen(\cdot), Rep(\cdot)\}$ with $\{Ctx^n_2, RGD^n_{U_x}, W^n_4, Gen(\cdot), Rep(\cdot)\}$ in its own database.

5. Security analysis

In this section we will demonstrate the resiliency of IoTD-2-CS and UX-2-CS against various security threats through informal and formal security analysis.

5.1. Informal security analysis

Informal security analysis refers to non mathematical security analysis, which is carried out in this section to elaborate the resiliency of IoTD-2-CS and UX-2-CS against various security attacks.

5.1.1. Replay attack

The prevention of replay attacks is imperative in the proposed AKA mechanism. In the proposed IoTD-2-CS and UX-2-CS AKA mechanisms, we use timestamps to prevent replay attacks. In the IoTD-2-CS AKA mechanism, there are three messages exchanged during the AKA phase: M_1, M_2, and M_3. Each message incorporates the latest timestamp. The validity of the timestamps is verified by the receiving network entity using the conditions $T_{dl} \geq |T_{re} - T_x|$, $T_{dl} \geq |T_{re} - T_y|$, and $T_{dl} \geq |T_{re} - T_z|$ for messages M_1, M_2, and M_3, respectively. If any of these conditions fail, the associated message is considered to be delayed. In the UX-2-CS AKA mechanism, the receiving network entity verifies the freshness of the timestamps by using the conditions $T_{dl} \geq |T_{re} - T_g|$, $T_{dl} \geq |T_{re} - T_h|$, and $T_{dl} \geq |T_{re} - T_i|$ for messages Mg_1, Mg_2, and Mg_3, respectively. If any of these conditions fail, the related message is regarded as delayed. In this way, both AKA mechanisms prevent replay attacks.

5.1.2. DoS attack

In this attack, it is desirable to prevent legitimate users from generating too many AKA messages, which could overwhelm the resources of the CS_y. These AKA messages are used to validate the user at CS_y. In the proposed UX-2-CS AKA mechanism, users must perform local authentication before generating an AKA message with the parameters $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$. To accomplish local authentication, U_x must check a specific condition $W_4 \stackrel{?}{=} W_3$. If this condition is met, U_x will generate Mg_1. Otherwise, U_x will be unable to generate Mg_1. In this way, the proposed UX-2-CS mechanism can prevent the DoS attack.

5.1.3. MITM attack

In the proposed IoTD-2-CS AKA mechanism, three messages are exchanged: $M_1 : \{T_x, Y_1, Y_2, Y_4\}$, $M_2 : \{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}$, and $M_3 : \{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}$. These messages can be intercepted as discussed in the threat model. An attacker, after capturing these messages, can alter and resend them to the user or cloud server to compromise the AKA phase. However, the attacker cannot fabricate these messages without possessing the secret credentials, such as ID_i, R_{x1}, CS_y long-term secret key, and random numbers. Furthermore, the integrity of M_1, M_2, and M_3 is verified through the conditions $Y_5 \stackrel{?}{=} Y_4$, $Y_{11} \stackrel{?}{=} Y_{10}$, and $Y_{15} \stackrel{?}{=} Y_{16}$. If any of these conditions fail, the AKA phase will be terminated. Similarly, the attacker cannot generate valid messages $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$, $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$, and $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$, communicated during the AKA phase of UX-2-CS, without the secret credentials used in the construction of these messages. Additionally, the integrity of Mg_1, Mg_2, and Mg_3 is checked through the conditions $W_6 \stackrel{?}{=} W_7$, $W_{12} \stackrel{?}{=} W_{13}$, and $W_{18} \stackrel{?}{=} W_{19}$, respectively. The AKA phase will be successful if all the conditions are met; otherwise, it will be terminated. In this way, both AKA mechanisms can prevent MITM attacks.

5.1.4. IoT device impersonation attack

In the proposed IoTD-2-CS AKA mechanism, there are two message exchanges that occur during the AKA phase: $M_1 : \{T_x, Y_1, Y_2, Y_4\}$, and $M_3 : \{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}$. These messages are carefully crafted using a set of secret parameters, including the IoT device's identity, the long-term secret key of the cloud server, K_d, challenge and response parameters, and a secret key derived from the FE's key reproduction function. Lacking possession of these crucial parameters, an attacker would be unable to produce valid M_1 and M_3 to successfully impersonate as IoT device. In this way, the proposed IoTD-2-CS AKA mechanism can prevent the IoT device impersonation attacks.

5.1.5. User impersonation attack

In UX-2-CS AKA mechanism, messages exchanges during the AKA phase: $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$ and $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$. Both messages are constructed using the secret parameters: the identity of the user, the long-term secret key of the cloud server, and the secret key generated from the FE's key reproduction function. Additionally, these messages incorporate fresh random numbers. Therefore, both temporary and long-term secret credentials are necessary to modify these messages. Without possessing these parameters, the attacker cannot generate valid Mg_1 and Mg_3 to impersonate the legitimate user. In this way, the proposed UX-2-CS AKA mechanism can prevent the user impersonation attacks.

5.1.6. Cloud server impersonation attack

In this attack, the attacker generates a random message with random parameters to make it appear to U_x as if it is from the legitimate cloud server. In the IoTD-2-CS system, the cloud server generates the message $M_2 : \{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}$ in response to M_1 received from the IoT device. The attacker cannot generate a valid message without having parameters such as K_d, ID_i, a random number, and a challenge response parameter stored in the cloud server's database. Therefore, the proposed IoTD-2-CS AKA mechanism can withstand cloud server impersonation attacks. Similarly, to generate the valid message $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$, which is transmitted by the cloud server during the AKA phase of the UX-2-CS AKA mechanism, an attacker would need to know the parameters K_2, R_i, ID_{U_x}, and the challenge response parameters stored in the cloud server's database. Since these parameters are known only to the user and the cloud server, the attacker cannot generate the valid message Mg_2. Therefore, the proposed UX-2-CS AKA mechanism effectively protects against cloud server impersonation attacks.

Table 3
ROR Model Queries.

Query	Explanation of the Query
$Execute(\Phi_{U_x}^{I_1}, \Phi_{CS_y}^{I_2})$	Using this query, an adversary can extract the exchanged messages during a legitimate execution of the protocol between a client instance and server instance.
$Test(\Phi^{I_1})$	Consider a bit $b \in 0, 1$ that has been picked at arbitrary. Upon asking this query, if b = 1, the output symbolizes the real secret value for instance U_x. However, if b = 0, the outcome consists of an arbitrary vector of the same size as the secret value. If the secret value for U_x is undefined, the query yields null value.
$Reveal(\Phi^{I_1})$	$\mathcal{A}$ can access the session key maintained by oracle Φ^{I_1} with this query.
$Send(\Phi^{I_1}, M)$	When communication is intercepted, an adversary might alter a message and reroute it to its intended recipient. This query produces the response message generated by instance U_x upon receiving message M.
$Corrupt(\Phi^{I_1})$	This query simulates a smart stolen device attack, where an adversary extracts all secret parameters stored in the smart device employing physical attacks, such as side-channel analysis.

5.1.7. De-synchronization attack

De-synchronization occurs due to the updating of one or more parameters during the AKA phase. However, it is possible that some parameters are updated on one network entity, while due to an eavesdropping attack, the parameters that need updating remain unchanged on the other side. In the proposed AKA mechanism, such a state does not exist, thereby preventing any desynchronization.

5.1.8. Password guessing attack

In this attack, the attacker's objective is to change or update the user's password after somehow obtaining the user's device. The attacker utilizes power analysis attacks to extract data stored in the device's memory, including $\{Ctx_2, RGD_{U_x}, W_4, Gen(\cdot), Rep(\cdot)\}$. With this information, the attacker can change the user's password only if the condition $W_4 \stackrel{?}{=} W_3$ is true. This condition holds true only if the attacker knows the user's secret credentials, such as the password, identity, and biometric key. Without these credentials, the condition $W_4 \stackrel{?}{=} W_3$ cannot be met. Therefore, the attacker cannot successfully execute a password change or password guessing attack against the proposed UX-2-CS AKA mechanism.

5.1.9. Temporary parameter leakage attack

This paper proposes two AKA mechanisms. In IoTD-2-CS, the session key $SK_i(= SK_y) = H(K_{d1} \parallel CH_i^n \parallel R_p^n \parallel R_n \parallel R_m)$ is generated during its AKA phase. During the AKA phase of UX-2-CS, the session key $SK_{U_x}(= SK_{CS_y}) = H(K_f \parallel CH_{U_x}^n \parallel R_{U_x}^n \parallel K_2 \parallel R_k \parallel R_i \parallel R_j)$ is generated. Both session keys are derived from a combination of temporary secret credentials and permanent credentials of various entities within the healthcare application. To compromise the security of either session key, an adversary must simultaneously compromise both the temporary and permanent credentials of the network entities. This ensures that the proposed AKA mechanisms can effectively temporary parameter leakage attack.

5.1.10. Anonymity and un-traceability attack

During the execution of the AKA phases of IoTD-2-CS and UX-2-CS AKA mechanisms the following messages are exchanged $M_1 : \{T_x, Y_1, Y_2, Y_4\}$, $M_2 : \{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}$, and $M_3 : \{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}$ and $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$, $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$, and $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$, respectively. All these messages are random due to the involvement of the current timestamps and random numbers. The adversary can not find any correlation between different AKA sessions. In addition, the attacker can't find the identity of the user and IoT device from the captured messages. Hence, the proposed AKA mechanisms provide anonymity and un-traceability features.

5.2. Security analysis using random or real (ROR) model

Through ROR model, UX-2-CS is thoroughly examined, and $\mathcal{A}$ is given permission to construct a variety of queries that allow for the execution of legitimate attacks. Various components of the ROR model are described as follows.

Participants: Within the UX-2-CS framework, three key entities/participants are involved: U_x (User), CS_y (Mobile Edge Server). The instances I_1 and I_2 representing U_x and CS_y are denoted as $\Phi_{U_x}^{I_1}$ and $\Phi_{CS_y}^{I_2}$, respectively, functioning as oracles.

Partnership: If instances $\Phi_{U_x}^{I_1}$ and $\Phi_{CS_y}^{I_2}$ have a common SK, they establish a partnership at the acceptance state.

Freshness: By $\mathcal{A}$, the SK generated during the AKA phase between $\Phi_{U_x}^{I_1}$ and $\Phi_{CS_y}^{I_2}$ cannot be revealed or made public.

Table 3 contains a list of these queries. We evaluate every potential query in order to formally verify the security of UX-2-CS. The subsequent variety of queries are used to simulate various attack scenarios against UX-2-CS.

Theorem 1. *Let $\mathcal{A}$ be a polynomial time (pti) bounded adversary challenging the security of UX-2-CS. We use HQ^2, qr_s, and HP^2 to represent hash, send, and PUF queries, respectively. The password dictionary space is denoted as $PASD$, and the length of the biometric key is indicated by 2^{le}. Moreover, the parameters C' and s' are defined in Zipf's law as described in [63]. $|PUS|$ and $|HS|$ denote the PUF*

and hash output space. Furthermore, $Adv_{\mathcal{A}}^{IND-CPA}(pti)$ represents the advantage of $\mathcal{A}$ in compromising the security of an AES-CBC. The estimation of $\mathcal{A}$'s advantage in compromising the security of the session key generated during the AKA phase of UX-2-CS is as follows.

$$Adv_{\mathcal{A}}^{UX-2-CS}(pti) \leq \frac{HQ^2}{|HS|} + \frac{HP^2}{|PUS|} + \max\left\{C' \cdot qr_s^{s'}, \frac{qr_s}{2^{le}}\right\} + 2 \cdot Adv_{\mathcal{A}}^{IND-CPA}(pti) \tag{73}$$

Proof. We establish the proof of Theorem 1 by analyzing the trailing five games GM_0, GM_1, GM_2, GM_3, and GM_4 for UX-2-CS [8]. The adversary $\mathcal{A}$'s advantage in compromising the security of the secret session key is denoted as $Adv_{\mathcal{A}}^{UX-2-CS}(pti) = |2 \cdot Adv^{GM} - 1|$. Here, "$Adv^{GM}$" signifies the likelihood of $\mathcal{A}$ winning by accurately predicting the bit "b" in each game.

GM_0: In this scenario, $\mathcal{A}$ launches a genuine attack on UX-2-CS AKA mechanisms. According to the success criteria, we have attained the intended result.

$$Adv_{\mathcal{A}}^{UX-2-CS}(pti) = |2 \cdot Adv^{GM_0} - 1|. \tag{74}$$

GM_1: In this game, $\mathcal{A}$ can intercept messages like Mg_1, Mg_2, and Mg_3 exchanged during AKA phase of UX-2-CS, through an eavesdropping attack facilitated by the query $Execute(\Phi_{U_x}^{I_1}, \Phi_{CS_y}^{I_2})$. After successfully intercepting Mg_1, Mg_2, and Mg_3, $\mathcal{A}$ must generate a valid session key $SK_{U_x}(= SK_{CS_y}) = H(K_f \| CH_{U_x}^n \| R_{U_x}^n \| K_2 \| R_k \| R_i \| R_j)$, which is created by combining both long-term and temporary random parameters. In the final phase of GM_1, $\mathcal{A}$ performs the operations $Reveal(\Phi^{I_1})$ to uncover the suspected secret key and $Test(\Phi^{I_1})$ to compare the actual secret key with a random bit. Without access to both the long-term and temporary random parameters, $\mathcal{A}$ cannot generate a valid session key. Therefore, the probability of $\mathcal{A}$ succeeding is considered negligible. As a result, GM_0 and GM_1 become indistinguishable. Hence, we can infer that:

$$Adv^{GM_1} = Adv^{GM_0} \tag{75}$$

GM_2: In this game, $\mathcal{A}$ employs a hash query to an oracle to launch an active attack. The hash function is utilized in the proposed UX-2-CS AKA mechanism for generating the encryption key and session key (SK) for the user and CS_y, respectively. Additionally, the hash function is involved in generating $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$, $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$, and $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$ during the AKA phase of UX-2-CS. By performing multiple hash searches, $\mathcal{A}$ aims to find collisions, thereby compromising the security of both the encryption key and SK. All communicated messages, such as Mg_1, Mg_2, and Mg_3 during the AKA phases of UX-2-CS, contain elements of unpredictability due to the inclusion of secret parameters, timestamps, random integers, and identities. However, the probability of finding collisions is considered negligible, as illustrated by the birthday paradox. This low likelihood of collision also applies to PUF queries, similar to hash queries.

$$Adv^{GM_2} - Adv^{GM_1} \leq \frac{HQ^2}{2|HS|} + \frac{HP^2}{2|PUS|}. \tag{76}$$

GM_3: During the course of the game, $\mathcal{A}$ initiated an active attack by executing the $Corrupt(\Phi^{I_1})$ query. Subsequently, upon successfully compromising the device used by the user, the attacker managed to obtain a set of credentials $\{Ctx_2, RGD_{U_x}, W_4, Gen(\cdot), Rep(\cdot)\}$ stored in the device's memory. $\mathcal{A}$'s objective is to ascertain the user's password. By making password guesses, $\mathcal{A}$ can validate them using the extracted information Ctx_2, RGD_{U_x} and W_4 by leveraging Zipf's law on passwords [63–67]. In scenarios involving trawling guessing attacks, $\mathcal{A}$'s success rate exceeds 0.5 when $qr_s = 10^7$ or 10^8. Moreover, when $\mathcal{A}$ employs the target user's personal data in targeted guessing attacks, $\mathcal{A}$'s success rate surpasses 0.5 when $qr_s \leq 10^6$. Additionally, since FE in UX-2-CS can extract a maximum of le random bits, the likelihood of $\mathcal{A}$ guessing the biometric key $\sigma_{U_x} \in \{0,1\}^{le}$ is approximately 2^{le} [58,68]. However, the likelihood of accurately guessing the biometric key is extremely low, approximately $\frac{1}{2^{le}}$, due to the inherent difficulty in guessing biometric data. Furthermore, the system imposes limitations on the number of failed password attempts allowed. If the system restricts the number of incorrect password attempts, Zipf's law on passwords leads to the following result:

$$Adv^{GM_3} - Adv^{GM_2} \leq \max\left\{C' \cdot qr_s^{s'}, \frac{qr_s}{2^{le}}\right\}. \tag{77}$$

GM_4: In this gaming scenario, $\mathcal{A}$ employs $Execute(\Phi_{U_x}^{I_1}, \Phi_{CS_y}^{I2})$ to capture three messages: $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$, $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}\}$, and $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$. All messages exchanged during the AKA phase of UX-2-CS are encrypted using the AES-CBC encryption algorithm. Upon obtaining these messages, $\mathcal{A}$ aims to uncover all confidential data that was encrypted and transmitted between the user and CS_y. To achieve this, $\mathcal{A}$ must compromise the security of the AES-CBC mechanism. The following objective is achieved:

$$Adv^{GM_4} - Adv^{GM_3} \leq Adv_{\mathcal{A}}^{IND-CPA}(pti) \tag{78}$$

Upon completion of all games, $\mathcal{A}$ fails to attain a considerable advantage in accurately forecasting the bit "b". Therefore, we conclude that

$$Adv^{GM_4} = 1/2 \tag{79}$$

From (74) and (75), we get

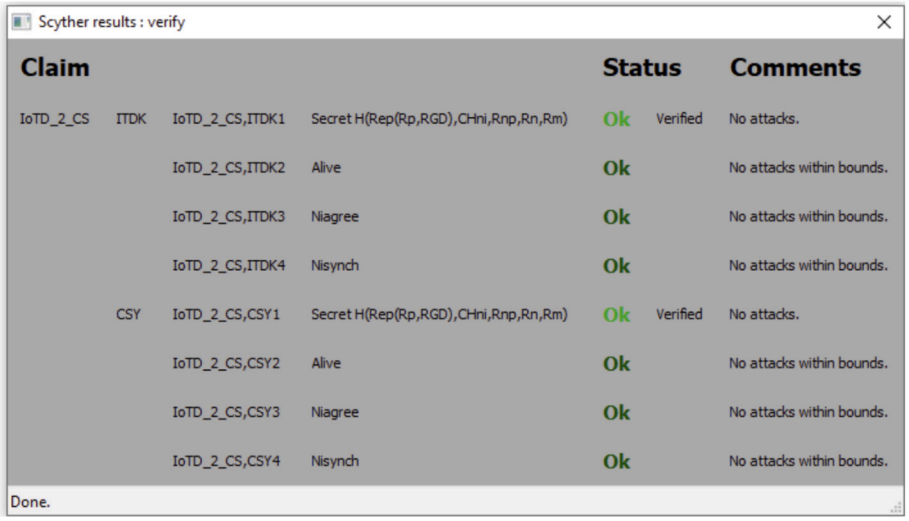


Fig. 5. Security analysis of IoTD-2-CS AKA mechanism using the Scyther tool.

$$Adv_{\mathcal{A}}^{UX-2-CS}(pti) = |\, 2 \cdot Adv^{GM_0} - \frac{1}{2} \,|\,. \tag{80}$$

From (80), we get

$$\frac{1}{2}.Adv_{\mathcal{A}}^{UX-2-CS}(pti) = |\, Adv^{GM_0} - Adv^{GM_4} \,|\,. \tag{81}$$

By using (79) and (81), we obtain

$$\frac{1}{2}.Adv_{\mathcal{A}}^{UX-2-CS}(pti) = |\, Adv^{GM_1} - Adv^{GM_4} \,| \tag{82}$$

Upon considering the triangular inequality, we have

$$\begin{aligned} |\, Adv^{I_1} - Adv^{GM_4} \,| \leq & |\, Adv^{GM_1} - Adv^{GM_2} \,| \\ & + |\, Adv^{GM_2} - Adv^{GM_4} \,| \\ \leq & |\, Adv^{GM_1} - Adv^{GM_2} \,| + |\, Adv^{GM_2} - Adv^{GM_3} \,| \\ & + |\, Adv^{GM_3} - Adv^{GM_4} \,|\,. \end{aligned} \tag{83}$$

By using (76), (78), and (83), we get

$$Adv_{\mathcal{A}}^{UX-2-CS}(pti) \leq \frac{HQ^2}{|\,HS\,|} + \frac{HP^2}{|PU|} + \max\left\{ C' \cdot qr_s^{s'}, \frac{qr_s}{2^{le}} \right\} + 2.Adv_{\mathcal{A}}^{IND-CPA}(pti). \quad \square \tag{84}$$

5.3. Security evaluation of IoTD-2-CS and the UX-2-CS AKA mechanisms using scyther

We utilized the Scyther tool to assess the security robustness of the proposed IoTD-2-CS and the UX-2-CS AKA mechanisms. Scyther employs the Security Protocol Description Language (SPDL) to express code, providing a reliable simulation environment. We opted for Scyther over AVISPA due to the several advantages it offers. Notably, it excels in identifying multi-protocol attacks, assumes the coexistence of multiple protocols on the same network, utilizes SPDL as its primary language, facilitates the identification of multi-protocol attacks, and supports both finite and unbounded session counts. Additionally, Scyther generates attack graphs if attacks are detected within the protocol and allows protocol assessment with a predetermined or infinite number of sessions. Scyther serves as an automated tool for evaluating, confirming, and assessing security frameworks and methods. Its unique features make it a valuable asset, publicly accessible for use. Particularly, Scyther's "pattern refinement algorithm" aids in providing concise representations of trace sets, aiding in categorizing potential protocol actions and security issues. Its widespread usage within research circles attests to its efficiency. In evaluating the IoTD-2-CS AKA and UX-2-CS AKA mechanisms, implementations are conducted using SPDL. Scyther evaluates the SPDL script representing two essential roles for UX-2-CS: U_x for the user and CS_y for the medical cloud server. Similarly, Scyther assesses the SPDL script representing two crucial roles for IoTD-2-CS: ITD_k for the IoT device and CS_y for the medical cloud server. The SPDL script, as illustrated in Fig. 5 and Fig. 6 comprises various claims associated with each role, all of which are verified by Scyther.

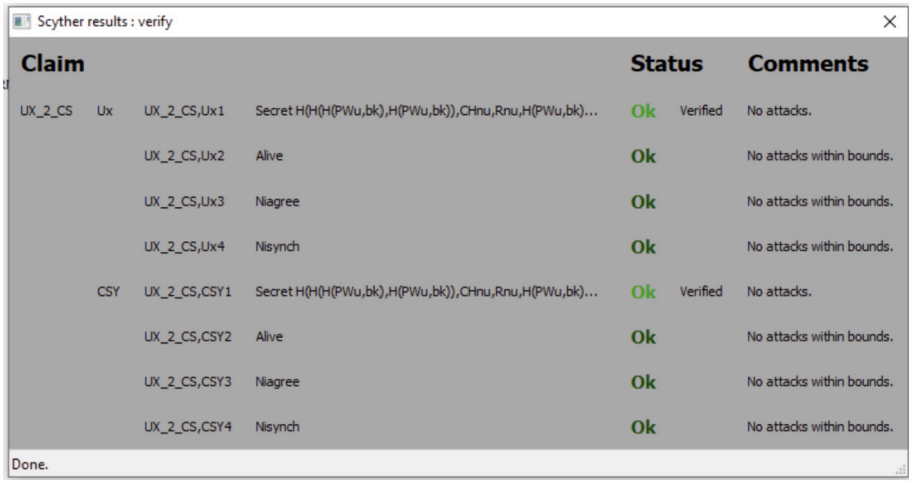


Fig. 6. Security analysis of UX-2-CS AKA mechanism using the Scyther tool.

Table 4
Average Computational Time for Cryptographic Primitives.

Cryptographic Primitive	Symbol	Average Computational Time (ms)
ECC Multiplication	T_{EM}	2.88
ECC Addition	T_{EA}	0.016
Hash Algorithms	T_{H}	0.309
Fuzzy Extractors	T_{FE}	2.88
Bi-linear Pairings	T_{BP}	32.08
Symmetric Encryption/Decryption	T_{ENC}	0.018/0.014
Physical Unclonable Functions	T_{PF}	0.00054 [71]
Exponential	T_{EX}	0.039

6. Results and performance analysis

We assess the efficiency of the proposed IoTD-2-CS AKA mechanism in terms of computational cost, communication overhead, and security functionalities. For simulating the cloud server, IoT and user devices, we employ Raspberry Pi 3 B+ Rev 1.3, running Ubuntu 20.04 LTS (64-bit OS), with a 1.4 GHz quad-core processor, 4 cores, and 1 GB of RAM. We use the cryptographic library "MIRACL" for implementing the all the cryptographic primitives used in the proposed IoTD-2-CS and UX-2-CS AKA mechanisms and related security schemes. The execution time for each primitive is provided in Table 4 [69,70].

6.1. Security features analysis

The security feature of the proposed IoTD-2-CS and UX-2-CS AKA mechanism are compared with the related AKA schemes. Table 5 provides a comparison of the security features between the proposed AKA mechanisms and their associated security schemes. It clearly shows that the proposed AKA mechanism offers enhanced security features, notably through the integration of the PUF function at the medical cloud server side. Moreover, the PUF is used to generate the secret key for the medical cloud server, which is not explicitly stored in the server's database. This effectively prevents insider attacks from accessing the secret key of the medical cloud server. This approach helps in preventing privileged insider attacks within the proposed AKA mechanisms, a guarantee not offered by other related security schemes.

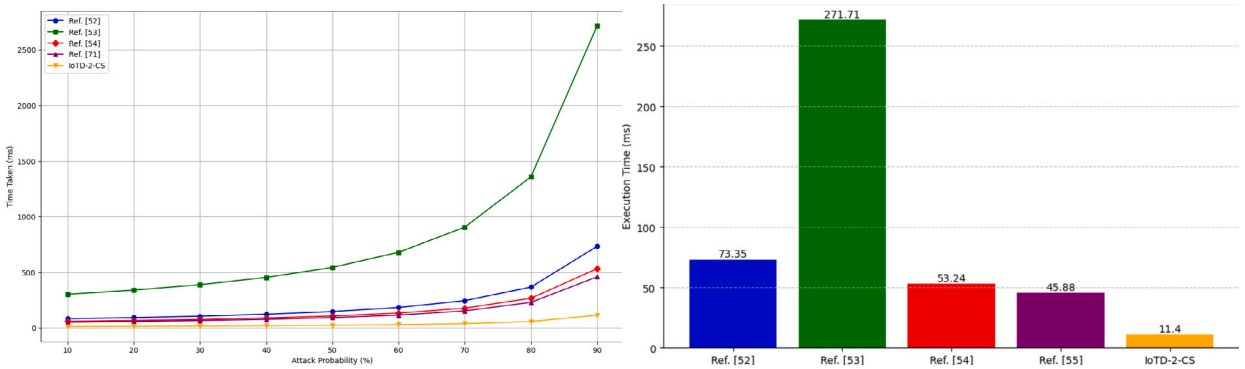
6.2. Computational cost analysis

In this subsection, we analyze the computational cost required to complete the IoTD-2-CS AKA mechanism and the UX-2-CS AKA mechanism. The computational cost of the IoTD-2-CS AKA mechanism is 14.4 ms, which is 84.46%, 95.80%, 78.59%, and 75.15% less than Ref. [52], Ref. [53], Ref. [54], and Ref. [55], respectively. A comparison of the computational cost between the IoTD-2-CS AKA mechanism and related security schemes is presented in Table 6 and Fig. 7(b). Similarly, the computational cost of the UX-2-CS AKA mechanism is 86.25%, 72.76%, 69.51%, and 71.06% less than Ref. [48], Ref. [49], Ref. [50], and Ref. [51], respectively. A comparison of the computational cost between the UX-2-CS AKA mechanism and related security schemes is provided in Table 6 and Fig. 7(d). During the AKA phase, communication happens through an open channel, making it vulnerable to various attacks such

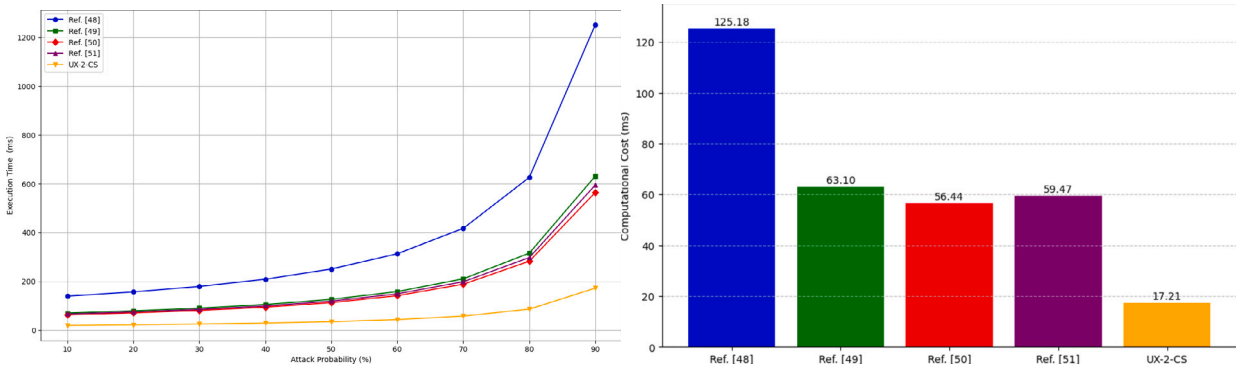
Table 5
Analysis of Security Function.

AKA Mechanism	*SFEA-A*	*SFEA-B*	*SFEA-C*	*SFEA-D*	*SFEA-E*	*SFEA-F*	*SFEA-G*	*SFEA-H*	*SFEA-I*
Ref. [48]	×	×	✓	✓	✓	✓	×	✓	-
Ref. [49]	✓	×	✓	✓	✓	✓	×	✓	-
Ref. [50]	✓	×	✓	✓	✓	✓	×	✓	-
Ref. [55]	✓	×	✓	✓	✓	✓	×	✓	-
Ref. [52]	✓	×	✓	✓	✓	✓	×	✓	✓
Ref. [53]	✓	×	✓	✓	✓	✓	×	✓	✓
Ref. [54]	✓	×	✓	✓	✓	✓	×	✓	✓
Ref. [51]	✓	×	✓	✓	✓	✓	×	✓	✓
UX-2-CS	✓	✓	✓	✓	✓	✓	✓	✓	✓
IoTD-2-CS	✓	✓	✓	✓	✓	✓	✓	✓	N/A

SFEA-A: "Impersonation Attack"; *SFEA-B:* "PUF functionality"; *SFEA-C*: "MITM Attack"; *SFEA-D*: "Anonymity"; *SFEA-E*: "Mutual Authentication"; *SFEA-F*: "Replay Attack"; *SFEA-G*: "Privileged Insider Attack", *SFEA-H*: "DoS Attack", *SFEA-I*: "Password Guessing Attack"; ✓: "reflects the function's supported"; ×: "signifies the not supported feature.".



(a): Unknown attack can cause the computational delay. (b): Computational cost comparison of AKA phase of IoTD-2-CS and other AKA mechanisms.



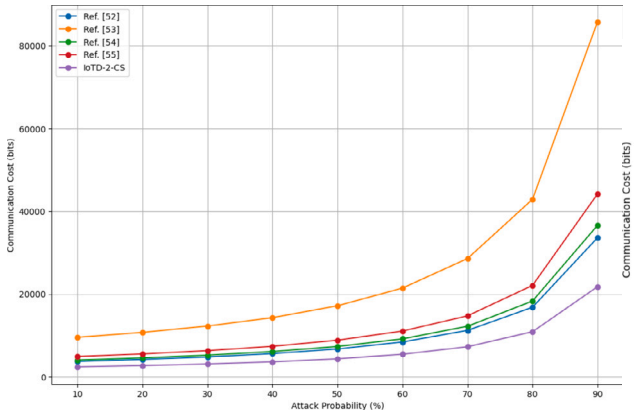
(c): Unknown attack can cause the computational delay. (d): Computational cost comparison of AKA phase of UX-2-CS and other AKA mechanisms.

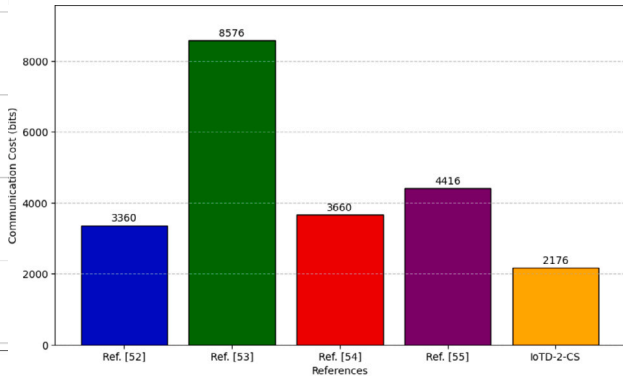
Fig. 7. Comparison of computational and communication costs.

as jamming and eavesdropping. These attacks can disrupt the execution of the AKA phase, leading to increased computational time. This scenario is illustrated in Fig. 7(a) and Fig. 7(c).

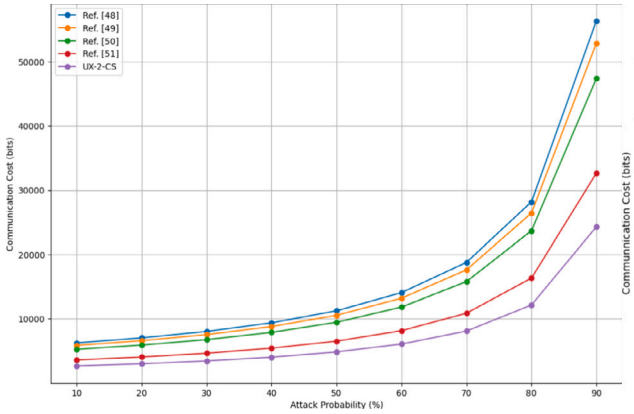
6.3. Communication cost analysis

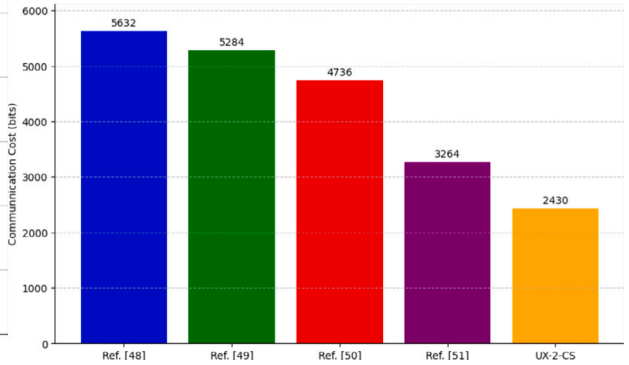
In this subsection, we estimate the communication of both the IoTD-2-CS AKA mechanism and the UX-2-CS AKA mechanism. To estimate the communication cost for both mechanisms, we assume the length of various parameters as follows: ECC point size is 320 bits, hash algorithm output length is 256 bits, random numbers are 128 bits, the AES-CBC block size is 128 bits, the AES-CBC key size is





(e): Unknown attack can cause the computational delay. (f): Computational cost of AKA phase of IoTD-2-CS and related AKA mechanisms.





(g): Communication cost comparison of UX-2-CS's AKA phase and related AKA mechanisms under unknown attacks. (h): Communication cost comparison of UX-2-CS's AKA phase and related AKA mechanisms.

Fig. 7. (*continued*)

Table 6
Computational Cost Analysis of IoTD-2-CS and UX-2-CS AKA Mechanisms.

AKA Mechanism	Computational Cost for UX-2-CS
Ref. [48]	$3T_{BP} + 9T_{EM} + 4T_{EX} + 5_{EA} + 9T_H \approx 125.81$ **ms**
Ref. [49]	$T_{BP} + 10T_{EM} + T_{EX} + T_{EA} + 7T_H \approx 63.10$ **ms**
Ref. [50]	$T_{BP} + 7T_{EM} + 2T_{EA} + 4T_{EX} + 13T_H \approx 56.44$ **ms**
Ref. [51]	$20T_{EM} + T_{EA} + 6T_H \approx 58$ **ms**
UX-2-CS	$17T_H + 4T_{FE} + 24T_{ENC} + 2T_{PF} \approx 17.21$ **ms**
AKA Mechanism	**Computational Cost for IoTD-2-CS**
Ref. [52]	$23T_{EM} + 23T_H \approx 73.35$ **ms**
Ref. [53]	$4T_{BP} + 47T_{EM} + 26T_H \approx 271.71$ **ms**
Ref. [54]	$13T_{EM} + 2T_{FE} + 4T_{EX} + 32T_H \approx 53.24$ **ms**
Ref. [55]	$14T_{EM} + 18T_H \approx 45.88$ **ms**
IoTD-2-CS	$8T_H + 3T_{FE} + 16T_{ENC} + 2_{PF} \approx 11.4$ **ms**

Table 7
Communication Cost Analysis of UX-2-CS and IoTD-2-CS AKA Mechanisms.

AKA Mechanism	Communication Cost (bits)
Ref. [48]	5632
Ref. [49]	5284
Ref. [50]	4736
Ref. [51]	3264
UX-2-CS	2430
AKA Mechanism	Communication Cost (bits)
Ref. [52]	3360
Ref. [53]	8576
Ref. [54]	3660
Ref. [55]	4416
IoTD-2-CS	2176

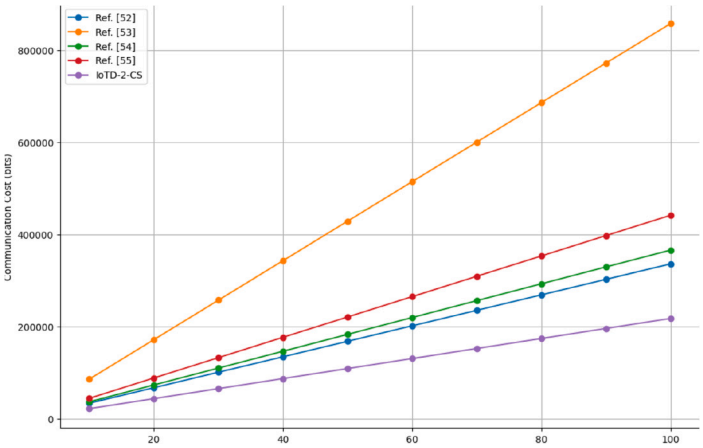


Fig. 8. Bandwidth requirement of IoTD-2-CS AKA mechanism and related AKA mechanisms.

256 bits, the initialization vector size is 128 bits, challenge size is 128 bits, response size is 128 bits, timestamp is of 32 bits size, identity size is 128 bits, and biometric key length is 256 bits. During the executing of the IoTD-2-CS AKA mechanism there are following message exchange to set up session key $M_1 : \{T_x, Y_1, Y_2, Y_4\}$, $M_2 : \{T_y, Y_6, Y_7, Y_8, Y_9, Y_{10}\}$, and $M_3 : \{T_z, RGD_i^*, Y_{12}, Y_{13}, Y_{14}, Y_{15}\}$. The sizes of M_1, M_2, and M_3 are 544 bits, 800 bits, and 832 bits, respectively, totaling 2176 bits to complete the AKA phase of the IoTD-2-CS AKA mechanism. In comparison, related security schemes such as Ref. [52], Ref. [53], Ref. [54], and Ref. [55] require 3360 bits, 8576 bits, 3660 bits, and 4416 bits, respectively. It is noteworthy that the IoTD-2-CS AKA mechanism entails 40.55%, 74.62%, 35.24%, and 50.72% less communication cost compared to schemes Ref. [52], Ref. [53], Ref. [54], and Ref. [55], respectively. Table 7 and Fig. 7(f) provide a comparative analysis of the communication costs during the IoTD-2-CS AKA mechanism. During the AKA phase, communication ensues via an open channel, making it susceptible to various attacks such as jamming and eavesdropping. These attacks can disrupt the execution of the AKA phase and increase the communication cost. This is illustrated in Fig. 7(e).

Similarly, during the execution of the UX-2-CS AKA mechanism, the following message exchanges occur to establish the session key: $Mg_1 : \{T_g, W_1, W_2, W_5, W_6\}$, $Mg_2 : \{T_h, W_8, W_9, W_{10}, W_{11}, W_{12}$, $Mg_3 : \{T_i, W_{12}, W_{13}, W_{14}, W_{15}, W_{16}\}$. The sizes of Mg_1, Mg_2, and Mg_3 are 800 bits, 800 bits, and 830 bits, respectively, totaling 2430 bits to complete the AKA phase of the IoTD-2-CS AKA mechanism. In contrast, related security schemes such as Ref. [48], Ref. [49], Ref. [50], and Ref. [51] require 5632 bits, 5284 bits, 4736 bits, and 3264, respectively. It's noteworthy that the UX-2-CS AKA mechanism entails 56.85%, 54.01%, 48.69%, and 25.55% less communication cost compared to schemes Ref. [48], Ref. [49], Ref. [50], and Ref. [51], respectively. Table 7 and Fig. 7(h) provide a comparative analysis of the communication costs during the UX-2-CS AKA mechanism. The AKA phase takes place over an open communication channel, making it vulnerable to attacks like jamming and eavesdropping. Such attacks can interfere with the AKA phase, resulting in higher communication costs. This scenario is depicted in Fig. 7(g).

Given the numerous users in the healthcare system accessing data from the cloud server simultaneously, along with many IoT devices collecting patient data and sending it to the cloud, it is essential to reduce bandwidth requirements during the AKA phase. The bandwidth requirements comparison between the proposed AKA mechanisms and related AKA mechanisms is shown in Fig. 8 and Fig. 9.

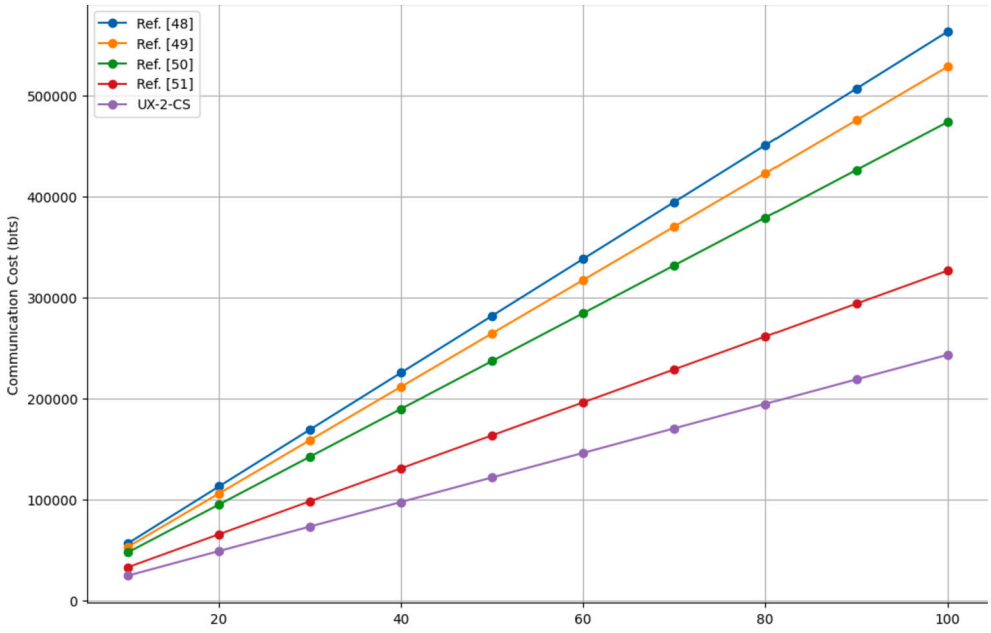


Fig. 9. Bandwidth requirement of UX-2-CS AKA mechanism and related AKA mechanisms.

7. Conclusion

Smart healthcare applications incorporate a mixture of IoT devices, wearables, sensors, and data analytics to oversee patient health, manage medical records, and facilitate remote consultations. Nevertheless, the implementation of these applications has raised significant security concerns. To meet the security requirements, this paper introduced two AKA mechanisms, IoTD-2-CS and UX-2-CS, which utilize physical unclonable functions, symmetric encryption, and hash functions to bolster security. In IoTD-2-CS and UX-2-CS, a session key was established between the IoT device, user, and cloud server through mutual authentication to enable encrypted communication. Informal security analysis demonstrated the resilience of these proposed mechanisms against various attacks. Furthermore, the efficiency of the AKA mechanisms was assessed against other security mechanisms, revealing that the UX-2-CS AKA mechanism required 69 to 86.25 percent less computational cost and 25.55 to 56.85 percent lower communication cost. Additionally, the IoTD-2-CS AKA mechanism for IoT devices required 75 to 95.80 percent less computational cost and 35.24 to 74.62 percent lower communication cost. These results advocate that the proposed AKA mechanisms are appropriate for healthcare applications.

CRediT authorship contribution statement

Omar Alruwaili: Writing – original draft, Validation, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Muhammad Tanveer:** Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Faisal Mohammed Alotaibi:** Visualization, Validation, Resources, Investigation, Funding acquisition, Conceptualization. **Waleed Abdelfattah:** Visualization, Supervision, Resources, Methodology, Investigation, Formal analysis. **Ammar Armghan:** Writing – review & editing, Visualization, Supervision, Project administration, Investigation, Funding acquisition, Data curation, Conceptualization. **Faeiz M. Alserhani:** Validation, Supervision, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Ammar reports a relationship with Al Jouf University that includes: employment. Ammar has patent n/a pending to n/a. The authors of this manuscript would like to declare that there are no conflicts of interest regarding the publication of this paper. We confirm that there has been no financial support or personal relationships that could have appeared to influence the work reported in this manuscript.

If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No external data is used in this paper, and all the data generated are part of the paper.

Acknowledgement

This work was funded by the Deanship of Graduate Studies and Scientific Research at Jouf University under grant No. (DGSSR-2023-02-02355).

References

[1] C.-M. Chen, Z. Chen, A.K. Das, S.A. Chaudhry, A security-enhanced and ultra-lightweight communication protocol for internet of medical things, IEEE Int. Things J. (2023).
[2] C.-M. Chen, Z. Chen, S. Kumari, M.S. Obaidat, J.J.P.C. Rodrigues, M.K. Khan, Blockchain-based mutual authentication protocol for iot-enabled decentralized healthcare environment, IEEE Int. Things J. (2024) 1, https://doi.org/10.1109/JIOT.2024.3396488.
[3] A. Srivastava, A. Kumar, Efficient Methods for Authentication for Internet-of-Things Devices Based on e-Health Scheme, Artificial Intelligence, Blockchain, Computing and Security, vol. 2, CRC Press, 2024, pp. 407–412.
[4] I.A. Abbasi, S.U. Jan, A.S. Alqahtani, A.S. Khan, F. Algarni, A lightweight and robust authentication scheme for the healthcare system using public cloud server, PLoS ONE 19 (1) (2024) e0294429.
[5] T. Arpitha, D. Chouhan, J. Shreyas, Anonymous and robust biometric authentication scheme for secure social iot healthcare applications, J. Eng. Appl. Sci. 71 (1) (2024) 8.
[6] V.K. Bathalapalli, S.P. Mohanty, E. Kougianos, V. Iyer, B. Rout, Pufchain 3.0: hardware-assisted distributed ledger for robust authentication in healthcare cyber–physical systems, Sensors 24 (3) (2024) 938.
[7] X. Chen, B. Wang, H. Li, A privacy-preserving multi-factor authentication scheme for cloud-assisted iomt with post-quantum security, J. Inf. Secur. Appl. 81 (2024) 103708.
[8] M. Tanveer, A.K. Bashir, B.A. Alzahrani, A. Albeshri, K. Alsubhi, S.A. Chaudhry, Cadf-cse: chaotic map-based authenticated data access/sharing framework for iot-enabled cloud storage environment, Phys. Commun. 59 (2023) 102087.
[9] C.-M. Chen, Z. Chen, S. Kumari, M.S. Obaidat, J.J. Rodrigues, M.K. Khan, Blockchain-based mutual authentication protocol for iot-enabled decentralized healthcare environment, IEEE Int. Things J. (2024).
[10] M. Tanveer, S.A. Chelloug, M. Ahmad, A.A. Abd El-Latif, et al., Leaf-iiot: lightweight and efficient authentication framework for the industrial internet of things, IEEE Access (2024).
[11] M. Tanveer, M. Ahmad, T.N. Nguyen, A.A. Abd El-Latif, et al., Resource-efficient authenticated data sharing mechanism for smart wearable systems, IEEE Trans. Netw. Sci. Eng. (2022).
[12] M.K. Hasan, Z. Weichen, N. Safie, F.R.A. Ahmed, T.M. Ghazal, A survey on key agreement and authentication protocol for internet of things application, IEEE Access (2024).
[13] Z.A. Hussien, H. Jin, Z.A. Abduljabbar, M.A. Hussain, A.A. Yassin, S.H. Abbdal, M.A. Al Sibahee, D. Zou, Secure and efficient e-health scheme based on the internet of things, in: 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2016, pp. 1–6.
[14] Y. Chen, Y. Ge, Y. Wang, Z. Zeng, An improved three-factor user authentication and key agreement scheme for wireless medical sensor networks, IEEE Access 7 (2019) 85440–85451, https://doi.org/10.1109/ACCESS.2019.2923777.
[15] D.S. Gupta, S.H. Islam, M.S. Obaidat, A. Karati, B. Sadoun, Laac: lightweight lattice-based authentication and access control protocol for e-health systems in iot environments, IEEE Syst. J. 15 (3) (2021) 3620–3627, https://doi.org/10.1109/JSYST.2020.3016065.
[16] M. Adeli, N. Bagheri, H.R. Maimani, S. Kumari, J.J.P.C. Rodrigues, A post-quantum compliant authentication scheme for iot healthcare systems, IEEE Int. Things J. 11 (4) (2024) 6111–6118, https://doi.org/10.1109/JIOT.2023.3309931.
[17] M. Wazid, A.K. Das, N. Kumar, J.J.P.C. Rodrigues, Secure three-factor user authentication scheme for renewable-energy-based smart grid environment, IEEE Trans. Ind. Inform. 13 (6) (2017) 3144–3153, https://doi.org/10.1109/TII.2017.2732999.
[18] S. Roy, S. Chatterjee, A.K. Das, S. Chattopadhyay, S. Kumari, M. Jo, Chaotic map-based anonymous user authentication scheme with user biometrics and fuzzy extractor for crowdsourcing internet of things, IEEE Int. Things J. 5 (4) (2018) 2884–2895, https://doi.org/10.1109/JIOT.2017.2714179.
[19] S.H. Islam, P. Vijayakumar, M.Z.A. Bhuiyan, R. Amin, V. Rajeev M., B. Balusamy, A provably secure three-factor session initiation protocol for multimedia big data communications, IEEE Int. Things J. 5 (5) (2018) 3408–3418, https://doi.org/10.1109/JIOT.2017.2739921.
[20] H. Zhu, X. Hao, A provable authenticated key agreement protocol with privacy protection using smart card based on chaotic maps, Nonlinear Dyn. 81 (2015) 311–321.
[21] Y. Liu, K. Xue, An improved secure and efficient password and chaos-based two-party key agreement protocol, Nonlinear Dyn. 84 (2016) 549–557.
[22] J.-L. Tsai, N.-W. Lo, T.-C. Wu, Novel anonymous authentication scheme using smart cards, IEEE Trans. Ind. Inform. 9 (4) (2012) 2004–2013.
[23] Q. Jiang, F. Wei, S. Fu, J. Ma, G. Li, A. Alelaiwi, Robust extended chaotic maps-based three-factor authentication scheme preserving biometric template privacy, Nonlinear Dyn. 83 (2016) 2085–2101.
[24] M. Tanveer, S.A. Chelloug, M. Alabdulhafith, A.A.A. El-Latif, Lightweight authentication protocol for connected medical iot through privacy-preserving access, Egypt. Inform. J. 26 (2024) 100474, https://doi.org/10.1016/j.eij.2024.100474, https://www.sciencedirect.com/science/article/pii/S1110866524000379.
[25] X. Li, J. Peng, M.S. Obaidat, F. Wu, M.K. Khan, C. Chen, A secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems, IEEE Syst. J. 14 (1) (2019) 39–50.
[26] M. Masud, G.S. Gaba, K. Choudhary, M.S. Hossain, M.F. Alhamid, G. Muhammad, Lightweight and anonymity-preserving user authentication scheme for iot-based healthcare, IEEE Int. Things J. 9 (4) (2021) 2649–2656.
[27] A.M. Koya, P. Deepthi, Anonymous hybrid mutual authentication and key agreement scheme for wireless body area network, Comput. Netw. 140 (2018) 138–151.
[28] A. Gupta, M. Tripathi, A. Sharma, A provably secure and efficient anonymous mutual authentication and key agreement protocol for wearable devices in wban, Comput. Commun. 160 (2020) 311–325.
[29] M. Tanveer, A. Alkhayyat, S.A. Chaudhry, Y.B. Zikria, S.W. Kim, et al., Reas-tmis: resource-efficient authentication scheme for telecare medical information system, IEEE Access 10 (2022) 23008–23021.
[30] P. Kumar, S.-G. Lee, H.-J. Lee, E-sap: efficient-strong authentication protocol for healthcare applications using wireless medical sensor networks, Sensors 12 (2) (2012) 1625–1647.
[31] D. He, N. Kumar, J. Chen, C.-C. Lee, N. Chilamkurti, S.-S. Yeo, Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks, Multimed. Syst. 21 (2015) 49–60.
[32] F. Wu, L. Xu, S. Kumari, X. Li, An improved and anonymous two-factor authentication protocol for health-care applications with wireless medical sensor networks, Multimed. Syst. 23 (2017) 195–205.

[33] J. Srinivas, D. Mishra, S. Mukhopadhyay, A mutual authentication framework for wireless medical sensor networks, J. Med. Syst. 41 (2017) 1–19.
[34] R. Amin, S.H. Islam, G. Biswas, M.K. Khan, N. Kumar, A robust and anonymous patient monitoring system using wireless medical sensor networks, Future Gener. Comput. Syst. 80 (2018) 483–495.
[35] R. Ali, A.K. Pal, S. Kumari, A.K. Sangaiah, X. Li, F. Wu, An enhanced three factor based authentication protocol using wireless medical sensor networks for healthcare monitoring, J. Ambient Intell. Humaniz. Comput. (2018) 1–22.
[36] M. Bayat, A.K. Das, M. Pournaghi, H.A.N. Far, M. Fotuhi, M. Doostari, A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care iot, Comput. Netw. Int. J. Comput. Telecommun. 1 (1) (2020).
[37] C.-M. Chen, Z. Li, S.A. Chaudhry, L. Li, Attacks and solutions for a two-factor authentication protocol for wireless body area networks, Secur. Commun. Netw. 2021 (2021) 1–12.
[38] D. He, S. Zeadally, Authentication protocol for an ambient assisted living system, IEEE Commun. Mag. 53 (1) (2015) 71–77.
[39] A. Aldosary, M. Tanveer, Paaf-shs: puf and authenticated encryption based authentication framework for the iot-enabled smart healthcare system, Int. Things 26 (2024) 101159, https://doi.org/10.1016/j.iot.2024.101159.
[40] H. Amintoosi, M. Nikooghadam, M. Shojafar, S. Kumari, M. Alazab, Slight: a lightweight authentication scheme for smart healthcare services, Comput. Electr. Eng. 99 (2022) 107803.
[41] A. Kumari, V. Kumar, M.Y. Abbasi, S. Kumari, P. Chaudhary, C.-M. Chen, Csef: cloud-based secure and efficient framework for smart medical system using ecc, IEEE Access 8 (2020) 107838–107852, https://doi.org/10.1109/ACCESS.2020.3001152.
[42] R. Hajian, S. ZakeriKia, S. Erfani, M. Mirabi, Shaparak: scalable healthcare authentication protocol with attack-resilience and anonymous key-agreement, Comput. Netw. 183 (2020) 107567, https://doi.org/10.1016/j.comnet.2020.107567, https://www.sciencedirect.com/science/article/pii/S1389128620312147.
[43] Z.-Y. Wu, Y.-C. Lee, F. Lai, H.-C. Lee, Y. Chung, A secure authentication scheme for telecare medicine information systems, J. Med. Syst. 36 (2012) 1529–1535.
[44] H. Debiao, C. Jianhua, Z. Rui, A more secure authentication scheme for telecare medicine information systems, J. Med. Syst. 36 (2012) 1989–1995.
[45] Q. Jiang, M.K. Khan, X. Lu, J. Ma, D. He, A privacy preserving three-factor authentication protocol for e-health clouds, J. Supercomput. 72 (2016) 3826–3849.
[46] W. Wang, Q. Chen, Z. Yin, G. Srivastava, T.R. Gadekallu, F. Alsolami, C. Su, Blockchain and puf-based lightweight authentication protocol for wireless medical sensor networks, IEEE Int. Things J. 9 (11) (2021) 8883–8891.
[47] J. Ryu, J. Oh, D. Kwon, S. Son, J. Lee, Y. Park, Y. Park, Secure ecc-based three-factor mutual authentication protocol for telecare medical information system, IEEE Access 10 (2022) 11511–11526, https://doi.org/10.1109/ACCESS.2022.3145959.
[48] A. Irshad, M. Sher, H.F. Ahmad, B.A. Alzahrani, S.A. Chaudhry, R. Kumar, An improved multi-server authentication scheme for distributed mobile cloud computing services, KSII Trans. Int. Inf. Syst. 10 (12) (2016) 5529–5552.
[49] Y. Li, Q. Cheng, X. Liu, X. Li, A secure anonymous identity-based scheme in new authentication architecture for mobile edge computing, IEEE Syst. J. 15 (1) (2020) 935–946.
[50] M. Rakeei, F. Moazami, An efficient and provably secure authenticated key agreement scheme for mobile edge computing, Wirel. Netw. 28 (7) (2022) 2983–2999.
[51] M. Seifelnasr, R. AlTawy, A. Youssef, E. Ghadafi, Privacy-preserving mutual authentication protocol with forward secrecy for iot-edge-cloud, IEEE Int. Things J. (2023).
[52] P. Porambage, A. Braeken, C. Schmitt, A. Gurtov, M. Ylianttila, B. Stiller, Group key establishment for enabling secure multicast communication in wireless sensor networks deployed for iot applications, IEEE Access 3 (2015) 1503–1511.
[53] B. Yu, H. Li, Anonymous authentication key agreement scheme with pairing-based cryptography for home-based multi-sensor internet of things, Int. J. Distrib. Sens. Netw. 15 (9) (2019) 1550147719879379.
[54] M. Wazid, A.K. Das, N. Kumar, M. Alazab, Designing authenticated key management scheme in 6g-enabled network in a box deployed for industrial applications, IEEE Trans. Ind. Inform. 17 (10) (2020) 7174–7184.
[55] K.-H. Wang, C.-M. Chen, W. Fang, T.-Y. Wu, A secure authentication scheme for internet of things, Pervasive Mob. Comput. 42 (2017) 15–26.
[56] M. Tanveer, A. Aldosary, S.-u.-d. Khokhar, A.K. Das, S.A. Aldossari, S.A. Chaudhry, Paf-iod: puf-enabled authentication framework for the internet of drones, IEEE Trans. Veh. Technol. (2024) 1–15, https://doi.org/10.1109/TVT.2024.3365992.
[57] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inf. Theory 29 (2) (1983) 198–208.
[58] M. Wazid, A.K. Das, V. Odelu, N. Kumar, M. Conti, M. Jo, Design of secure user authenticated key management protocol for generic iot networks, IEEE Int. Things J. 5 (1) (2017) 269–282.
[59] M. Wazid, A.K. Das, V. Odelu, N. Kumar, W. Susilo, Secure remote user authenticated key establishment protocol for smart home environment, IEEE Trans. Dependable Secure Comput. 17 (2) (2017) 391–406.
[60] S. Yu, K. Park, Puf-based robust and anonymous authentication and key establishment scheme for v2g networks, IEEE Int. Things J. 11 (9) (2024) 15450–15464, https://doi.org/10.1109/JIOT.2024.3349689.
[61] C. Tian, J. Ma, T. Li, J. Zhang, C. Ma, N. Xi, Provably and physically secure uav-assisted authentication protocol for iot devices in unattended settings, IEEE Trans. Inf. Forensics Secur. 19 (2024) 4448–4463, https://doi.org/10.1109/TIFS.2024.3379861.
[62] W. Hou, Y. Sun, D. Li, Z. Guan, J. Liu, Lightweight and privacy-preserving charging reservation authentication protocol for 5g-v2g, IEEE Trans. Veh. Technol. 72 (6) (2023) 7871–7883, https://doi.org/10.1109/TVT.2023.3241324.
[63] D. Wang, H. Cheng, P. Wang, X. Huang, G. Jian, Zipf's law in passwords, IEEE Trans. Inf. Forensics Secur. 12 (11) (2017) 2776–2791.
[64] D. Wang, Z. Zhang, P. Wang, J. Yan, X. Huang, Targeted online password guessing: an underestimated threat, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, pp. 1242–1254.
[65] S. Roy, A.K. Das, S. Chatterjee, N. Kumar, S. Chattopadhyay, J.J. Rodrigues, Provably secure fine-grained data access control over multiple cloud servers in mobile cloud computing based healthcare applications, IEEE Trans. Ind. Inform. 15 (1) (2018) 457–468.
[66] D. Wang, P. Wang, Two birds with one stone: two-factor authentication with security beyond conventional bound, IEEE Trans. Dependable Secure Comput. 15 (4) (2016) 708–722.
[67] Q. Wang, D. Wang, C. Cheng, D. He, Quantum2fa: efficient quantum-resistant two-factor authentication scheme for mobile devices, IEEE Trans. Dependable Secure Comput. 20 (1) (2021) 193–208.
[68] J. Srinivas, A.K. Das, N. Kumar, J.J.P.C. Rodrigues, Tcalas: temporal credential-based anonymous lightweight authentication scheme for internet of drones environment, IEEE Trans. Veh. Technol. 68 (7) (2019) 6903–6916, https://doi.org/10.1109/TVT.2019.2911672.
[69] B. Bera, A.K. Das, A.K. Sutrala, Private blockchain-based access control mechanism for unauthorized uav detection and mitigation in internet of drones environment, Comput. Commun. 166 (2021) 91–109.
[70] B. Bera, S. Saha, A.K. Das, A.V. Vasilakos, Designing blockchain-based access control protocol in iot-enabled smart-grid system, IEEE Int. Things J. 8 (7) (2020) 5744–5761.
[71] T. Alladi, N. Naren, G. Bansal, V. Chamola, M. Guizani, SecAuthUAV: a novel authentication scheme for UAV-ground station and UAV-UAV communication, IEEE Trans. Veh. Technol. 10 (2020), https://doi.org/10.1109/TVT.2020.3033060.