Journal of Systems Architecture 120 (2021) 102307

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

PUA-KE: Practical User Authentication with Key Establishment and its Application in Implantable Medical Devices

Neha Kumari [a], Jayaprakash Kar [a,*], Kshirasagar Naik [b]

[a] Center for Cryptography, Cyber Security & Digital Forensics, Department of Computer Science & Engineering, The LNM Institute of Information Technology, Jaipur, India

[b] Department of Electrical and Computer Engg, University of Waterloo, Waterloo, Ontario, N2L3G1, Canada

ARTICLE INFO

Keywords:
Authentication
Key-establishment
Controller
Implementable Medical Device

ABSTRACT

The use of Implantable Medical Devices (IMDs) in the arena of medical sciences have provided a quantum leap in network transformation by permitting and retrieving the technology on demand. However, with the constant progress of these devices with respect to wireless communication and the potential for outside caregiver to communicate wirelessly have increased its impact to security, and infringement in privacy of human beings. The Controller Device (CD) is one of the most important component in the IMD communication network. The patient's vitals are stored in a medical server through this device. The IMD monitors the physical phenomena such as heart rate and blood glucose level of the patient and transmits the information to the CD through some type of wireless communication media like Bluetooth, WiFi etc. Then the CD forwards the information to the cloud server using a access point. The data is stored for further analysis and decision making by the medical expert such as doctor, healthcare provider for the consultation. The users (doctors, patients, or the emergency response team) can access the data stored in the cloud server after a successful authentication. In this paper, we propose an efficient and lightweight secure authenticated key establishment protocol (PUA-KE). The security is proven in random oracle model and the experimental analysis demonstrates that the proposed scheme achieves a lower computational time and communication overhead at 80-bit, 112-bit and 128-bit security level in comparison to existing such schemes.

1. Introduction

The development of IMDs has offered human-beings an evolutionary way to treat major diseases which include diabetes, cardiac arrhythmia, cochlear, gastric diseases etc. With the rapid integration of computing devices and health care, many patients have started relying more on IMDs for treating their medical conditions and ailments. IMDs are electronic devices surgically implanted within the patient's body to treat a medical condition by monitoring the state for improving the functioning of a body part. They provide the patients with a capability that they did not possess before and enable remote monitoring of a patient's health status. These devices are small-sized to offer patients mobility without overloading them with heavy weights and also decrease the influence on the adjacent organs. IMDs are built-in with sensors which collect a range of physiological values such as heart rate, blood pressure, oxygen saturation, temperature, or neural activity etc. These values are being collected by the nearby CD using wireless communication technologies such as Bluetooth, infrared transmission etc. CD is a powerful device as compared to IMDs because it has high processing power and storage capability. The stored information is forwarded to the cloud server using the access point. The data is stored for further analysis and decision making by the medical expert, such as a doctor for the consultation. However, due to this wireless communication environment, patient's sensitive data can be intruded, changed, or deleted which makes these IMDs vulnerable to threats like data gather, trailing the patient, im-persuasion, relaying attacks and denial of service attack [1]. These threats violate confidentiality, integrity and accessibility properties of these devices, with a spotlight on the access management schemes to forestall unauthorized access [2]. These security attacks can even disrupt the battery of the embedded IMDs, as a result of which patient needs a surgical treatment for replacing the IMDs or in worst case, these attack might end up the patient in a shock state. Hence, security is a major concern for preventing such attacks. Proper authentication and access control mechanisms should be implemented in place to secure the IMDs. Any user like doctor, relative/friend of a patient, or the emergency response team should

* Corresponding author.
E-mail address: jayaprakashkar@lnmiit.ac.in (J. Kar).

https://doi.org/10.1016/j.sysarc.2021.102307
Received 8 May 2021; Received in revised form 4 October 2021; Accepted 6 October 2021
Available online 14 October 2021

only be allowed to access the patient's data on the Controller Device after successful authentication.

1.1. Motivation and contribution

The IMDs are designed with a number of sensors which collect the data related to physiological behaviors of patients. These includes the heart rate, blood pressure, oxygen saturation, temperature, or neural activity etc. These values are then forwarded to the nearby CD using wireless communication technologies such as Bluetooth, infrared transmission etc. The CD then forwards the information to the cloud server using the access point. The data is stored for further analysis and decision making by the healthcare provider. The users here is *i*.e the healthcare provider which includes doctors, relatives/friends of patients, emergency response team. Whenever he/she wants to access the data stored in the cloud server a successful authentication is required. So we are most concerned this security requirement. User authentication is an one of the most essential security requirement in the security of IMDs. This work presents a novel user authentication and key establishment protocol in certificateless setting. We put across the following contributions to satisfy the above security requirements.

(1) First we have proposed an efficient and lightweight user authentication scheme and key establishment protocol in random oracle model.
(2) We have proven the security of the proposed scheme using Client and Server security model.
(3) We have done an empirical analysis and shown that, the scheme achieves relatively less computational cost with respect to time and communication overhead to other schemes that are relatively pertinent to IMDs.
(4) We have evaluated the communication overhead and tested that the proposed scheme provides substantially less communication overhead at 80-bit, 112-bit and 128-bit security level.

1.2. Organization

Section 2 presents the related works where we have discussed user authentication and key establishment protocol proposed by the numerous of author based on identity-based cryptography (IBC) and certificateless public-key cryptography (CL-PKC). In Section 3 describes the hardware design of IMDs. Section 4 discusses the network model, operations and mathematical preliminaries. Section 5 presents the proposed scheme (PUA-KE) and the security model. The performance analysis has been done and describe in Section 6. Finally, in Section 7, we conclude our work.

2. Related works

In this section, we briefly present the previous works done on the security and privacy of Implantable medical devices. There are numerous of cryptographic protocols/primitives have been developed to achieve the security goal confidentiality, integrity, non-repudiation and authentication [2][3]. Many of the security mechanisms are based on public-key cryptography which establish a common secret between the IMD and the programmer, but this type of authentication may possess certain barriers related to computational complexity and energy consumption [4]. Therefore to overcome this the IMD security ought to either be supported lighter-weight and low-priced interchangeable encryption authentication schemes or the device itself be secured to mediate communication between an IMD and an external software engineer [3]. Several techniques have been proposed for authentication of IMDs, each of them has its own security weaknesses. Numerous of authentication schemes [5][6,7][8] have been proposed for the healthcare systems using radio-frequency identification (RFID), wireless medical sensor networks and wireless body area networks. To provide secure tele-healthcare service and monitoring health, He and Zeadally [9][10] proposed an authentication scheme for an Ambient Assisted Living (AAL) system using the ambient intelligence. It uses timestamps to avoid replay attacks. Mutual authentication is also guaranteed through the use of private keys, knowledge of which is unavailable to an adversary, while computing the request/response messages. In addition, the scheme ensures user anonymity and forward secrecy [3]. However these schemes are not efficient in computational time and communication overhead. Xu *et al.*. [11] proposed two cryptographic schemes establishment and access control for implantable cardiac devices known as IMDGuard . Rasmussen *et al.* [12] proposed a scheme based on ultrasonic distance bounding that allows the user to access the IMD within certain distance bounds set by the IMD. The security keys are generated after the proximity of the user has been verified, so the scheme is safe from any attack outside the proximity range. However, an attacker can mimic the ultrasonic signal used for proximity verification by inducing a current in the audio-receiver circuit, making it vulnerable to impersonation attack. Jang *et al.* [13] suggested a hybrid scheme that combined symmetric and asymmetric heterogeneous cryptosystems to ensure confidentiality and security against impersonation attacks by making use of bio-metric parameters and physiological data of the patient. A remote user authentication and session key agreement protocol was developed by Ravanbakhsh and Nazari for health care systems [14]. However, this scheme is not secure against session-specific temporary information attack and have lack of perfect forward secrecy. Later on Arezou *et al.* [15] proposed a similar user authentication and key agreement scheme for telecare medicine information systems (TMIS) preserving anonymity and unlinkablility based on Elliptic Curve Cryptosystem(ECC). But the computational cost is very high and is not secure against many attacks against server physical capture attack. An another mutual authentication scheme for TMIS with key establishment technique was proposed by Suresh Kumar *et al.* [16]. The protocol is analyzed against many security threats informally and using the formal method BAN logic. The protocol has been proven to be mutually authenticated. However the scheme has huge computational cost and does preserve user's anonymity.

Recent work [17,18] has shown that the wireless property is risky and can compromise with the confidentiality of the IMD's transmitted information. In different systems, designers use cryptographic strategies to supply confidentiality and forestall unauthorized access [19]. Therefore, adding cryptographic mechanisms on to IMDs is tough for an explicit range of reasons [3]. (i) In-alterable: Having cryptographic mechanisms and incorporating them in the IMDs may be infeasible due to limited memory space. (ii) Safety: It is a important point that the access of IMDs to be with the medical advisor. If the cryptographic mechanisms are embedded into IMDs,it may lead to situation where the access may be denied to a health care provider(doctor)due to invalid credentials [20]. (iii) Maintainability: Software package bugs are significantly problematic for IMDs as a result of they will cause device recollects. Such recollects are pricey and will need surgery if the model is already ingrained. Thus, it is recommended to limit IMDs' software package to solely medically necessary functions. Certain protocols such as by Darji and Trivedi [21] using a external proxy device that the patient wears, that conjointly shares a key with the IMD. During emergencies, however, access is exclusively provided by the IMD, the external device simply alerts the IMD. Even though the protocol has certain benefits like it conserves IMD battery power and addresses emergency authentication, the IMD is not accessible if the proxy device is lost or taken, that is undesirable throughout emergencies. Certain protocols earlier used bio-metric approach such as In 2011, Hei and Du [10] proposed a scheme for providing authentication at two levels for access control. Fingerprints and eye color are used at the first level, at the second level, a snapshot of the patient's iris is needed to check for authorization. However, IMDs must be pre-deployed with the patient's bio-metric information, which can be revealed after prolonged usage which can compromise the security of the patient. Public Key

Cryptography permits organization to line up a secure channel of transmission with no data concerning the previous keys. Each user generates a combination of keys referred to as public and private key. This mechanism solves the problem of key distribution and reduces the quantity of needed crypto-keys and shifts it concern towards the matter of key distribution to the matter of binding user together with his key combine. To bind the user with the key, Digital Certificates are used . Public-key infrastructure (PKI) proves legitimacy of users' keys by that of certificates. But it will possess sure drawbacks like the infrastructure is heavy-weight and privacy. Moreover, certificates should be verified by users to visualize whether or not they match with the right identity or not. Another disadvantage is revocation of old/compromised keys. Certificateless cryptography (CL-PKE) was introduced as a noteworthy various to ancient PKI. The access control scheme in our design follow certificateless cryptography . It makes use of identities, that are users' public keys shaped of absolute strings, in place of the certificates. It is conjointly light-weight and might be deployed at a lot of lower price. It conjointly offers clear cryptography, so non-technical users might simply secure their information. Siddiqi *et al.* [22] proposed a secure protocol for IMDs where the author presented a comprehensive security protocol for a modern IMD ecosystem, IMDfence, which addresses crucial, yet previously ignored requirements *i*.e non-repudiation, remote monitoring and system scalability. Also suggested a realistic solution for accessing the IMD during emergencies without compromising security or patient safety. In this a rigorous evaluation of the scheme has been performed paying special attention to the protection against battery denial-of-service (DoS) attacks. It is observed that this proposed protocol increases the total IMD energy consumption. Subsequently the same author Siddiqi *et al.*[23] introduced a securing IMD using Ultrasound Waves. A proof-of-concept implementation and validation of the Secure Echo approach has been illustrated and a comprehensive security evaluation of ultrasound as an inherently secure BCC channel has been discussed. The protocol discussed a lightweight device-pairing security that utilizes ultrasound in order to protect against battery-depletion attacks.

3. Hardware architecture

A medical Implantable device or Wearable device is configured with multiple of sensors. The device includes a host micro-controller, a safety co-processor and an actuator. A standard Hardware system model is presented in Fig. 1. The generic design of medical Implantable device embody a collection of sensors ingrained within the body of patient and use numerous parameters to judge the health-status and confirm the essential medical aid. All of the measurements are processed by the processing unit which is 32-bit CPU. Our Architecture functionalities of the Hardware model are summarized below:

- Memory: It is non-volatile storage. It is responsible of storing the collected measurements, all the previous health information and bound abnormalities, it additionally includes the medical care settings that are hold on in it and therefore are often retrieved by the server.
- Wireless identification and sensing platform (WISP): it is a engulfed device that pulls its energy from the incoming RF signal and might be browse by ultra-high-frequency identification (RFID) readers. It includes a completely programmable sixteen bit small controller that is able to execute scientific discipline algorithms by mistreatment the harvested energy.
- Programmer :TA programmer is employed to speak wirelessly with the ingrained devices, to speak with the device you wish to own a programmer equipped with a programming head to interrogate the device. Both the programmer and device communicate through a wireless association.
- MICS transceiver: It permits wireless communication with the IMD technologist. The MICS transceiver 10 chiefly consists of 3 elements (i) a four hundred megahertz transceiver, (ii) 2.45 rate receiver, and (iii) media access controller. It operates below the 402 – 405 megahertz MICS band 11.
- Processing unit: This unit is designed to execute and managing communication between the IMD and the programmer.

4. Preliminaries

This section briefs the network model which involves various entities participating in the protocol and its operation. The user's authentication and key establishment works on the IMD communication environment. Then mathematical foundations is presented by defining admissible bilinear mapping.

4.1. Network model

The Fig. 2 depicts the network model for the (IMD)s communication system. The model is build with various IMDs includes brain neurosimulator, gastric simulator which are implanted in a patient's body. A controller device CD which collects data from the neighboring IMDs using wireless communication technologies such as Bluetooth, zigbee and infrared transmission. CD is connected to the internet VIA an access point. The user such as health care providers, doctors etc. can access IMDs through CD. Let there is a user U would like to access the data from the controller device belonging to a set of IMDs. In this communication process, authentication is required between the user U and the controller device CD. The IMDs communication network involves the following components and entities.

- `Registration Center` : This is also known as Key Generation Centre KGC which is a trusted third party (TTP) also called registration authority. The KGC serves the registration for the users, IMDs and controller device. First the user has to register through his unique identity. This generates the partial private key for both the users and controller devices. The full private keys of user's and controller's are garnered by combining the secret parameters chosen at random by the respective entities and the partial private keys received from the KGC. Note that the KGC will send the partial private key to the respective users through secure channel using TLS protocol. Hence, our model does not suffer key escrow and certificate management overload. The registration and authentication process is depicted in Fig. 3.
- `Patient`: A patient is implanted with many IMDs such as pacemakers, a foot-drop implant,defibrillators, neuro simulator, a cochlear implant, a gastric simulator and an insulin pump. These IMDs have built-in sensors which are capable of collecting various psychological values (for example heart rate, blood pressure, temperature etc.).
- `Controller Device` (CD) : It is a medical sever which stores the psychological values sensed by the IMDs. These values are transferred to the nearest controller via wireless communication technologies *e*.g bluetooth, zigbee and infrared transmission. Here, the controller device acts as data owner and assume that the server is curious and honest. It has high computational capability and storage as compared to IMDs and other devices involves in the communication networks.
- Cloud Server: A server can be hired to store the data received from the CDs through the access point, where the data is being used for further analysis and decision making by an medical expert/healthcare provider such as a doctor or medical specialists in case consultation is required.
- `Users`: Users include Doctor, patient's relatives or any emergency service like ambulance etc can access the data stored in the could server after a successful authentication .

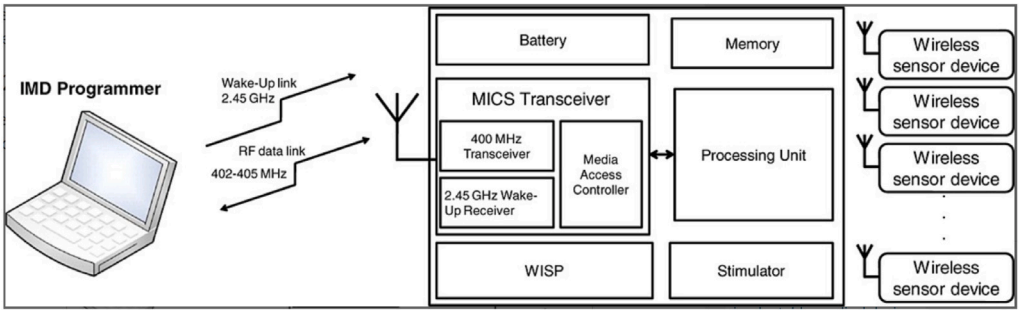

Fig. 1. Hardware Architecture.

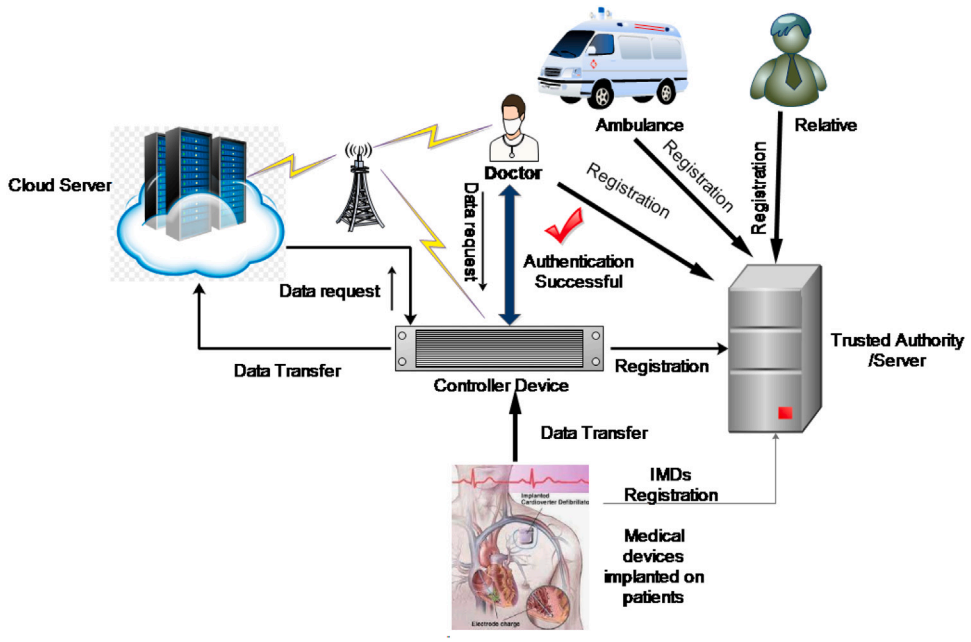

Fig. 2. Network model.

4.2. Authentication process and operation

In this section, we outline the authentication which are required between the various components and entities participated in the IMDs communication network. As suggested by He *et al.* [3] and Gollakota *et al.* [4], the IMDs environment requires following type of authentications.

- Prior to sending the data from IMD_k to CD_j, authentication is required between IMD_k and CD_j.
- Let the healthcare provider such as doctors is the user U_i would like to access the data stored in the cloud server CS_l through the access point, prior to access the data, authentication is required between U_i and CS_l.
- Let the user U_i say doctor would like to access the patent's real-time live data, authentication is required between IMDs and U_i. This will be performed through CS_l which act as a gateway between IMD_k and U_i.

We propose the authentication scheme PUA-KE that works between U_i and the controller device say CD_j. After a successful authentication, they establish a session key for further encryption and decryption process to establish a secure communication and to preserve privacy in the data. Finally, only U_i can access the patient's real-time live data from the implanted IMDs in the patient's body through CD_j.

4.3. Mathematical preliminaries

In this section, we define the mathematical preliminaries that have been used to design the protocol.Table 1 depicts the nomenclature for the notations used. In addition, we have specified the mathematical hard problems and assumption on which the security of our protocol relies. Let us consider two groups G_1 and G_2 of same order q. Where G_1 is an additive group and G_2 is a cyclic multiplicative group. A bilinear mapping $\hat{e}$ is defined as $\hat{e} : G_1 \times G_1 \rightarrow G_2$. The bilinear mapping is said to be admissible if it satisfies the following:

(1) Bilinearity: $\hat{e}(aP, bQ) = hate(P, Q)^{ab}$, where $P, Q \in G_1$ an d $a, b \in \mathbb{Z}_q^*$.
(2) Non-degeneracy: $\hat{e}(P, Q) \neq 1$ with given $P, Q \in G_1$ and 1 is the identity element in G_2.
(3) Computability: There exist an algorithm that can compute $\hat{e}(P, Q)$ efficiently for any $P, Q \in G_1$.

5. Proposed scheme

5.1. Initialisation phase

In this phase, the trusted authority (TA) acts as Key Generation enter (KGC) and sets an additive group G_1 and a multiplicative group G_2, both of same prime order p. Let P be the generator of group G_1. KGC

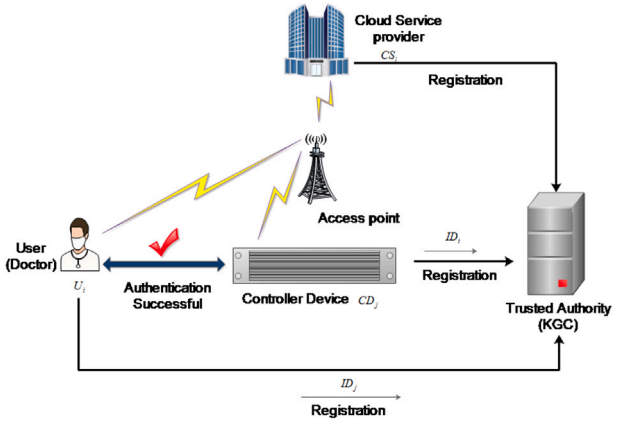

Fig. 3. User registration and Authentication.

Table 1
Nomenclature.

Notation	Meaning
υ	Security parameter
q	a large prime number
ID_i and ID_j:	User's and Server's identity respectively
G_1	additive cyclic group of order q
G_2	multiplicative cyclic group of same prime order q
$\hat{e}$	An admissible bilinear map
s	Master secret key
P_{pub}	Master Public key
pk_i and pk_j	User's and Server's Public key respectively
H_i, $i = 1,2,3,4$	Collision resistant hash function
D_i and D_j	User's and Server's Partial Private key
ψ	Expiration date
TS	Time stamp
$\perp$:	Null value

then sets four collision resistant hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $H_2 : G_1 \rightarrow \mathbb{Z}_p^*$, $H_3 : G_2 \rightarrow \{0,1\}^*$ and $H_4 : \{0,1\}^* \times \{0,1\}^* \times G_1 \times G_2 \rightarrow \mathbb{Z}_p^*$ and a bilinear mapping $\hat{e} : G_1 \times G_1 \rightarrow G_2$. Then it randomly chooses $s \in \mathbb{Z}_p^*$ and computes $P_{pub} = sP$ and $g = \hat{e}(P,P)$. KGC publishes the system global parameters $\{G_1, G_2, p, P, H_1, H_2, H_3, H_4, P_{pub}, g\}$ and keeps s as the secret which is the master secret key.

5.2. Registration phase

In this phase, TA acts as a Registration Center (RC) and is responsible for registering each user (for example, doctor, relative) which is denoted by U prior to allow to access any patient's implantable medical devices IMD which are deployed in the controller. The protocol works as client and server setting. The registration phase performs the following operations.

(1) Both the user and Server have to register with their respective identities with RC. Let the user and server are denoted by U and S respectively. Their corresponding identities are ID_i and ID_j. Note that the identity is the unique number such as IP address or any allotted number provided to the device (user or server).
(2) After receiving ID_i, RC sets an expiration date for the user denoted by ψ. Let for instance the value of $\psi = \}\}2020 - 04 - 30\varepsilon$. On the given user's identity ID_i, RC computes partial private key $D_i = \frac{1}{H_1(ID_i \| \psi)+s} P$ and sends (ψ, D_i) to the respective user U through a secured channel using TLS protocol. Similarly when RC receives the server's identity ID_j, computes the partial private key $D_j = \frac{1}{H_1(ID_j)+s} P$ and sends D_j to the server.
(3) The two entities namely user and the server they are participating in the protocol construct their public and private key pairs through KGC. Note that both the entities do not have the full right to construct their private keys. In order to construct their full private keys, they needs to register with RC. The generation of key pairs performs the following computations.

(a) U chooses $x_i \in \mathbb{Z}_p^*$ at random and computes the public key $pk_i = x_i(H_1(ID_i \| \psi)P + P_{pub})$ and full private key as $sk_i = \frac{1}{x_i + H_2(pk_i)} D_i$ on given D_i.
(b) Similarly S chooses $x_j \in \mathbb{Z}_p^*$ at random and computes the public key $pk_j = x_j(H_1(ID_j)P + P_{pub})$ and full private key as $sk_j = \frac{1}{x_j + H_2(pk_j)} D_j$ on given D_j.

5.3. Authentication phase

The protocol allows the controller to authenticate the users who are being participated in the protocol. If the authentication gets success, then the user will allow to access the controller device and use the patient's real-time live data. The protocol proceeds as

(1) The user U chooses a random $k \in \{0,1\}^w$ of string size w and sets as the session key. Then chooses $\mu \in \mathbb{Z}_p^*$ at random and computes $\alpha = g^{\mu}$.
(2) Sets $h_1 = H_3(\alpha)$.
(3) Computes $C = (k \| TS \| ID_i \| \psi) \oplus h_1$, where TS denotes the time stamp, which is used to resist reply attack.
(4) Compute $h = H_4(k \| TS \| ID_i \| \psi \| \alpha)$.
(5) Compute $V = (\mu + h)sk_i$.
(6) Compute $Z = \mu\{pk_j + H_2(pk_j)(H_1(ID_j)P + P_{pub})\}$.

Note that in order to achieve user's anonymity, the protocol encrypts the user's identity by using symmetric key encryption scheme. The user U sends (C, V, Z) to the controller device S. After receiving (C, V, Z), the S performs the following

(1) Compute $\alpha = \hat{e}(Z, sk_j)$.
(2) Compute $h_1 = H_3(\alpha)$.
(3) Compute $C \oplus h_1 = (k \| TS \| ID_i \| \psi)$.
(4) Compute $h = H_4(k \| TS \| ID_i \| \psi \| \alpha)$.
(5) Checks if $\alpha = \hat{e}(V, pk_i + H_2(pk_i)(H_1(ID_i)P + P_{pub}))g^{-h}$.
(6) Chooses $k^* \in \{0,1\}^{w_1}$ at random and computes the cryptographic checksum or tag $\beta_1 = MAC_{k \oplus k^*}(TS)$.

If the verification equation holds, then the server S accepts U's access request. A session key k is established between U and S for future cryptographic operations. Note that the key is only known to U and S which ensures privacy for future communications. After establishing the session key, the server computes the cryptography checksum $\beta_1 = MAC_{k \oplus k^*}(TS)$ and sends (β_1, k^*) to U. Upon receiving the tag and k^*, U computes tag $\beta_2 = MAC_{k \oplus k^*}(TS)$ and checks if $\beta_2 = \beta_1$. If both are equal, then U is ensured that S knows the session key $k \oplus k^*$.

5.4. Revocation phase

This phase is responsible for automatically revoking the privileges of user after the expiration date assigned by the Key Generation Center (KGC). The KGC broadcasts the user's, the expiration date to all the Controller devices (CDs). All the CDs maintain a database to store a list of identities of revoked users, so that they may check before granting the privilege to any user.

5.5. Proof of correctness

$$\begin{aligned} \alpha &= \hat{e}(Z, sk_j) \\ Z &= \mu\{pk_j + H_2(pk_j)(H_1(ID_j)P + P_{pub})\} \\ &= \mu\{ x_j(H_1(ID_j)P + P_{pub}) + H_2(pk_j)(H_1(ID_j)P + P_{pub})\} \\ &= \mu(x_j + H_2(pk_j))(H_1(ID_j)P + P_{pub}) \end{aligned}$$

$= \mu P(x_j + H_2(pk_j))(H_1(ID_j) + s)$

Since, $sk_j = \frac{1}{x_j + H_2(pk_j)} \cdot \frac{1}{H_1(ID_j)+s} \cdot P$

Hence, $\hat{e}(Z, sk_j) = \hat{e}(\mu P, P) = \hat{e}(P, P)^{\mu} = \alpha$.

We need to check the verification equations

$\alpha = \hat{e}V, pk_i + H_2(pk_i)(H_1(ID_i P + P_{pub}))g^{-h}$.

$$\hat{e}(V, pk_i + H_2(pk_i)(H_1(ID_i)P + P_{pub}))g^{-h} =$$
$$\hat{e}(\mu + h)\frac{1}{x_i + H_2(pk_i)} \frac{1}{H_1(ID_i \parallel \psi) + s} P, x_i(H_1(ID_i \parallel \psi)P + P_{pub})$$
$$+H_2(pk_i)(H_1(ID_i P + P_{pub}))g^{-h}.$$
$$= \hat{e}((\mu + h)\frac{1}{x_i + H_2(pk_i)} \frac{1}{H_1(ID_i \parallel \psi) + s} P,$$
$$x_i P(H_1(ID_i \parallel \psi) + s) + H_2(pk_i)(H_1(ID_i)P + P_{pub}))g^{-h}$$
$$= \hat{e}((\mu + h)\frac{1}{x_i(H_1(ID_i \parallel \psi) + s) + H_2(pk_i)(H_1(ID_i \parallel \psi) + s)} P,$$
$$x_i P(H_1(ID_i \parallel \psi) + s) + H_2(pk_i)H_1(ID_i)P + P_{pub})g^{-h}$$
$$= \hat{e}((\mu + h)P, P)g^{-h}$$
$$= \hat{e}(P, P)^{(\mu+h)}g^{-h} = g^{\mu} = \alpha$$

5.6. Security and adversary model

In this section, we have discussed the adversary model and proven the security. Authentication ensures the communicating party is one that it claims. Non-repudiation prevents the denial of previous commitments or actions. The cryptographic primitive encryption can provide the privacy and digital signature can ensures the integrity, authentication, and non-repudiation. Many cryptographic application needs ensure simultaneously all the security goals confidentiality, integrity, authentication, and non-repudiation. Thus, traditional mechanism is first perform the signing on a given message and then encrypt, called the sign-then-encrypt or signature then encryption approach. A new cryptographic primitive was proposed by Zheng [24] in 1987. This provides both the functions of digital signature and public key encryption simultaneously. This leads very lower computational cost than the traditional mechanism signing then encryption.

The proposed **PUA-KE** scheme works via the cryptographic primitive Signcryption which performs both signing and encryption in one single logical steps. So the security notion of **PUA-KE** is based on the signcryption describe in Section 5.3 which ensure confidentiality and unforgeability under the mathematical hard problem bilinear Diffie–Hellman inversion (BDHI) and q-Strong Diffie–Hellman problem(q-SDHI) respectively.

We prove the security of the proposed **PUA-KE** using the security model considering two communicating entities Client and Server describe in [25][26]. The security notion is defined as follows:

- The model consists of a set of participants that are modeled by some oracles. Let us we define the notation $\prod_{i,j}^{\pi}$. This means the user i believes that it carries out the π^{th} execution of the protocol with user j.
- The model consists of an adversary $\mathscr{A}$ that participate in the protocol and is allowed to access all message flows in this system. We assume that $\mathscr{A}$ can alter, modify, relay and delete the message. Note that $\mathscr{A}$ should not be one of the participant. Also it cannot be act as the entity KGC or RC.
- All the oracles establish their communication with each other via $\mathscr{A}$.
- The oracle maintains a number of transcripts that stores all messages they have sent and received during the communications.
- The adversary $\mathscr{A}$ cannot alter the message in the communication. Only can transmit the message. It means this act as passive adversary.

The simulation works in the following manner. We assume that $\mathscr{A}$ can submit a serious of polynomial number of bounded queries in an adaptive manner. This is described as

- `Create`: The adversary $\mathscr{A}$ selects an identity ID and set-up a user as new participant. The participant's key pairs are constructed by the help of KGC as describes in the protocol.
- `Send` : $\mathscr{A}$ chooses a message and transmits to an oracle i, $\prod_{i,j}^{\pi}$. Note that the message is being received from the participant j. Additionally, $\mathscr{A}$ can provide the instruction to participant j that to start a new execution of this protocol with i by sending a null message ϕ. If the first message that is received by the oracle is null message i.e $m = \phi$ then the oracle is called as **Initiator Oracle**, else the oracle is called **Responder Oracle**
- `Reveal`: $\mathscr{A}$ is allowed to ask an oracle to extract the session key if it holds.
- `Corrupt`: $\mathscr{A}$ is allowed to ask an oracle to extract the long term full private key.

The oracle has one of the possible several states:

(1) `Accepted`: The oracle has to decide to accept or reject the session key when the message is received .
(2) `Rejected`: If the session key is not set-up by the oracle, then it decides to reject and abort the simulation.
(3) If none of these above that means whether to accept or reject then the state of the oracle is assigned by the symbol $*$.
(4) The state is opened, if the oracle has answered to a reveal query.
(5) The state is corrupted, if the oracle has answered to an corrupted query.

So the simulation proceeds in the following ways:

- The adversary $\mathscr{A}$ selects one of these oracle $\prod_{i,j}^{\pi}$ and submit a `Test` query.
- The chosen oracle should be `accepted`, `unopened` and none of the participant i or j are corrupted.
- Additionally, there does not exists any opened oracle $\prod_{j,i}^{\pi}$ with having a matching conversation.

The oracle flips a fair coin $\xi \in \{0, 1\}$ to response a query. If $\xi = 0$, then the oracle answers the contains kept session key. Else the oracle chooses a random key from the key space and answers the query. The game is played between the adversary $\mathscr{A}$ and the Challenger $\mathscr{C}$. In this simulation, $\mathscr{A}$ submits polynomially bounded number of queries of `Create`, `Send`, `Reveal` and `Corrupt` with one `Test` query. At the end of this experiment, $\mathscr{A}$ produces a guessing bit ξ' for ξ. So the $\mathscr{A}$'s advantage is given by

$$\text{Adv}(\mathscr{A}) = |\Pr[\xi' = \xi] - \frac{1}{2}|$$

Definition 1. An authenticated key agreement protocol is said to be secure if it satisfies the following conditions.

(1) If there exists a passive attacker on the oracles $\prod_{i,i}^{\pi}$ and $\prod_{j,i}^{\pi}$, then the oracles must have the same session keys.
(2) If the two uncorrected oracles $\prod_{i,i}^{\pi}$ and $\prod_{j,i}^{\pi}$ have matching conversation for each adversary, then the oracles possessing the session keys are same.
(3) The advantage of the adversary $\mathscr{A}$ i.e $\text{Adv}(\mathscr{A})$ is negligible.

Now, we have proven the security of our protocol **PUA-KE** is secure with respect to the above definition.

Theorem 1. *The proposed* **PUA-KE** *protocol is a secure key establishment protocol.*

Proof. The proposed **PUA-KE** protocol is consistent and the session key is chosen at random from the key space $\{0, 1\}^l$. Thus it satisfies

the first condition. Since the protocol has consistency, the uncorrected oracles $\prod_{i,i}^{\lambda}$ and $\prod_{j,i}^{\theta}$ have matching conversation accepts the session key. Further the session key possesses by the oracles are session key. This holds the second condition. We can prove the third condition by the method of contradiction. We adopt the proof techniques use in [27]. Let us assume that there exists an adversary $\mathscr{A}$ has non-negligible advantage *epsilon* against the security illustrated in Definition 1. We can construct an algorithm $\mathscr{C}_1$ break the unforgeability of **PUA-KE** with an non-negligible advantage $\epsilon_1' \leq \frac{1}{uv}(\epsilon - \epsilon_2' uv\varphi)$. Similarly, we can construct an algorithm $\mathscr{C}_2$ break the confidentiality of **PUA-KE** with an non-negligible advantage $\epsilon_2' \leq \frac{1}{uv}(\epsilon - \epsilon_1' uv\varphi)$. This has been proven in [27]. We have proven this by the following two parts:

Part-I

Let us consider there are u number of clients with identities $\{ID_1, ID_2 \ldots ID_u\}$ and v number of servers $\{S_1, S_2 \ldots S_v\}$ number of servers participate in the protocol. We assume that let the all servers and the clients are activated at most γ and δ times by the adversary $\mathscr{A}$. So the time taken to break the confidentiality by the algorithm $\mathscr{C}_1$ is given by

$$t_1' \leq t_1 + u\delta t_c + v\gamma t_s$$

Where t_s and t_u denotes the response time for signcryption and unsigncryption queries respectively. t_1 denotes the time taken by $\mathscr{A}$ to distinguish the random value from a session key. Note that the algorithm $\mathscr{C}_1$ runs $\mathscr{A}$ as subroutine and is act as $\mathscr{A}$'s challenger. $\mathscr{C}_1$'s goal is to construct a valid signcryption ciphertext (C^*, V^*, Z^*) while the client with identity ID_A communicates with the server S_B. The algorithm $\mathscr{C}_1$ takes the system parameters and all v server's key pairs as input. Note that the algorithm does not take S_B's private key as input. $\mathscr{C}_1$ wins the game if and only if the challenged session key chosen by $\mathscr{A}$ is equal to the session key that is established by S_B and ID_A. In the simulation, $\mathscr{A}$ submits a polynomially bounded number of following queries and $\mathscr{C}_1$ sores all the state information in the state list Ls.

- `Create`: $\mathscr{A}$ submits create query with identity ID_i. Where ID_i is chosen by $\mathscr{A}$. So $\mathscr{A}$ submits the query with identity ID_i to key extraction oracle and obtains the private key sk_i. Note that if $ID_i = ID_A$, $\mathscr{A}$ fails to obtain the private key, since, the key extraction oracle cannot return a correct private key.
- `Corrupt`: $\mathscr{A}$ submits `Corrupt` query with identity ID_i or S_j. If $ID_i \neq ID_A$, $\mathscr{C}_1$ returns the private key sk_i, since it is known to him. If $S_j \neq S_B$, $\mathscr{C}_1$ returns the answer, since $\mathscr{C}_1$ knows the private key sk_j. Else $\mathscr{C}_1$ fails and abort the simulation. $\mathscr{C}_1$ must have to set the state of corrupted oracle as `Corrupted`.
- `Send`: This query is from `Create` and `Corrupt`. $\mathscr{A}$ submits the `Send` queries for all parities except S_B and ID_A. If $\mathscr{A}$ submits the `Send` query which is not associated with ID_A and S_B then $\mathscr{C}_1$ reply directly, since their private keys are known to him. In order to submit the `Send` query, $\mathscr{A}$ requires private keys of both ID_A and S_B. $\mathscr{C}_1$ uses his own oracle and returns the answer. When $\mathscr{A}$ submits the `Send` ($\prod_{A,i}^{\lambda}$) query with input a null message ϕ, $\mathscr{C}_1$ chooses a session key k at random and calls to signcryption oracle with input message $(k \parallel TS \parallel ID_i \parallel \psi)$ and public key pk_j to generate the ciphertext $\eta = (C, V, Z)$. $\mathscr{C}_1$ returns *eta* to $\mathscr{A}$ and update the list L_s with $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, *)$. When $\mathscr{A}$ submits the `Send`($\prod_{B,i}^{\lambda}$) query with input a ciphertext η with identity ID_i. If it returns $(k \parallel TS \parallel ID_i \parallel \psi)$ by calling the unsigncryption oracle, $\mathscr{C}_1$ marks the oracle as $\prod_{B,i}^{\lambda}$ as accepted, update the list L_s with $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Accepted`$)$ and returns the cryptographic checksum $\beta_1 = MAC_k(TS)$ to $\mathscr{A}$. The session is not accepted if it returns an output the symbol $\perp$ for failure. $\mathscr{C}_1$ returns `Rejected` to $\mathscr{A}$ and insert $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Rejected`$)$ into L_s. When $\mathscr{A}$ submits the `Send`($\prod_{A,i}^{\lambda}$) query with input the cryptographic checksum β_1, $\mathscr{C}_1$ searches the tuple $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, *)$ in list L_s, if it does not found then $\mathscr{C}_1$ marks as rejected to the oracle $\prod_{A,i}^{\lambda}$ and insert the tuple $(\lambda, A, j, \perp, \perp,$ `Rejected`$)$ in the list L_s. Else $\mathscr{C}_s$ again computes the cryptographic checksum $\beta_2 = MAC_k(TS)$ and check $\beta_1 = \beta_2$ or not. If it does not match, $\mathscr{C}_1$ marks as rejected to the oracle $\prod_{A,i}^{\lambda}$ and updates $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, *)$ as $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Rejected`$)$. If it matches, then the list is updated as $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, *)$ as $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Accepted`$)$
- `Reveal`: When $\mathscr{A}$ submit the `Reveal` query on the oracle $\prod_{i,j}^{\lambda}$. $\mathscr{C}_1$ searches the tuple (λ, i, j) in L_s for the accepted session key. The list L_s must contain the tuple (λ, i, j). Else it is an invalid query. When $\mathscr{C}_1$ found the tuple, it returns the corresponding session key and update $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Accepted`$)$ into $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Opened`$)$. At the end of the simulation, $\mathscr{C}_1$ searches the entry $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta,$ `Accepted`$)$ such that no tuple contains (λ, A, B). if it is true, then $\mathscr{C}_1$ returns its forge message $\eta^* = \eta$.

Let E denotes the event occurs when $\mathscr{A}$ submits the `Send`($\prod_{A,B}^{\lambda}$) query with input a signcryption ciphertext eta^* having the following conditions

- η^* is a valid ciphertext that is generated with input ID_A and pk_B.
- When submitting the `Send`($\prod_{A,B}^{\lambda}$) query with input a null message *phi*, this does not return η^* as the response.

If the event E occurs, then $\mathscr{C}_1$ founds the challenged tuple in L_s and returns the output "forgery" successfully. If $\mathscr{A}$ aborts the simulation and stop without selecting the test session between ID_A and S_B or the event does not occur then $\mathscr{C}_1$ fails. The probability of selecting test session when ID_A act as initiator and is S_B as responder is $\frac{1}{uv}$. Therefore $\mathscr{C}_1$'s advantage is

$$\text{Adv}(\mathscr{C}_1) = \epsilon_1' \geq \frac{Pr[E]}{uv}.$$

$\mathscr{C}_1$ calls the signcryption oracle to answer `Send`($\prod_{A,j}^{\lambda}$) query. The maximum number of such queries submitting is $u\delta$. Additionally, $\mathscr{C}_1$ calls the un signcryption oracle to answer `Send`($\prod_{B,i}^{\lambda}$) queries. The maximum number of such queries submitting is $v\gamma$. Therefore $\mathscr{C}_1$ forges the ciphertext in time

$$t_1' \leq t_1 + u\delta t_c + v\gamma t_s$$

Part-II In the second part, we have proven that when the event E does not occur. We assume that there does not exist any adversary $\mathscr{A}$ that can distinguish the random value chosen and the session key in time t_2. The goal of $\mathscr{C}_2$ is to break the confidentiality. Therefore, we construct an algorithm $\mathscr{C}_2$ which can forges in time

$$t_2' \leq t_2 + u\delta t_c + v\gamma t_s$$

The algorithm takes the system parameters *param*, master secret key s, key pairs of v servers $\{S_1, S_2 \ldots S_v\}$ as input. Note that it does not take the S_B's private key. The users participate in the protocol are $\{ID_1, ID_2 \ldots ID_u\}$. $\mathscr{C}_2$ uses $\mathscr{A}$ as subroutine and returns a random message $(k \parallel TS \parallel ID_i \parallel \psi)$ to its challenger. The challenger provides a challenged ciphertext η^*. The algorithm $\mathscr{C}_2$ chooses $\tau \in \{1, 2, \ldots \delta\}$ at random and guess selection of $\mathscr{A}$ of the challenged session. $\mathscr{C}_1$ maintains a list L_s that stores the state information. In the simulations $\mathscr{C}_2$ replays the queries that are being submitted by $\mathscr{A}$. $\mathscr{C}_2$ can reply to all queries because he knows private keys of all oracles except S_B. This is describes below:

- `Create`: $\mathscr{A}$ submits create query with identity ID_i. Where ID_i is chosen by $\mathscr{A}$. $\mathscr{C}_2$ uses the master secret key s and compute the partial private key D_i and the full private key sk_i. Here $\mathscr{A}$ is used as subroutine.
- `Corrupt`: $\mathscr{A}$ submits `Corrupt` query with identity ID_i or S_j. If $S_j \neq S_B$, $\mathscr{C}_1$ returns the private key sk_j, since it is known to him. Else $\mathscr{C}_2$ fails and abort the simulation. $\mathscr{C}_2$ must have to set the state of corrupted oracle as `Corrupted`.

- Send: This query is from Create and Corrupt. $\mathscr{A}$ submits the Send queries for all parities except S_B and ID_A. If $\mathscr{A}$ submits the Send query which does not involve S_B then $\mathscr{C}_2$ reply directly, since their private keys are known to him. In order to submit the Send query, $\mathscr{A}$ requires private key S_B. $\mathscr{C}_2$ uses his own oracle and returns the answer. When $\mathscr{A}$ submits the Send ($\prod_{B,i}^{\lambda}$) query with input signcryption ciphertext *eta*, $\mathscr{C}_2$ calls to unsigncryption oracle with input η and identity ID_i. If it produces the message $(k \parallel TS \parallel ID_i \parallel \psi)$, then $\mathscr{C}_2$ marks the oracle as $\prod_{B,i}^{\lambda}$ as accepted. Insert $(\lambda, B, i, k \parallel TS \parallel ID_i \parallel \psi, \eta, \texttt{Accepted})$ in L_s and returns the cryptographic checksum $\beta_1 = MAC_k(TS)$ to $\mathscr{A}$. The session is not accepted if it returns an output the symbol $\perp$ for failure. $\mathscr{C}_2$ returns Rejected to $\mathscr{A}$ and insert $(\lambda, B, i, k \parallel TS \parallel ID_i \parallel \psi, \eta, \texttt{Rejected})$ into L_s.
- Reveal: When $\mathscr{A}$ submit the Reveal query on the oracle $\prod_{i,j}^{\lambda}$. $\mathscr{C}_2$ searches the tuple (λ, i, j) in L_s. If it founds then it returns the session key k. $\mathscr{C}_1$ returns Rejected to $\mathscr{A}$ and insert $(\lambda, A, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, \texttt{Rejected})$ into L_s and update $(\lambda, i, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, \texttt{Accepted})$ into $(\lambda, i, j, k \parallel TS \parallel ID_i \parallel \psi, \eta, \texttt{Opened})$. Else the query is marked as invalid. When $\mathscr{A}$ submits Send($\prod_{A,B}^{\tau}$) query with input a null message *phi*, $\mathscr{C}_2$ chooses two messages k_0 and k_1 of equal length and call t challenged oracle with $k_0 \| TS \| ID_i \| \psi, k_1 \| TS \| ID_i \| \psi$. It obtains the challenged signcryption ciphertext η^* and returns to $\mathscr{A}$. $\mathscr{A}$ selects ($\prod_{A,B}^{\tau}$) session or a matching session that is created by Send($\prod_{A,B}^{\tau}$) query with input the signcryption ciphertext η^* as its target session. When $\mathscr{A}$ selects both the target and matching session, then $\mathscr{C}_2$ returns k_0 and k_1 to $\mathscr{A}$. At the end of the game, $\mathscr{A}$ returns the guess bit ϖ. If $\varpi = 0$, then $\mathscr{C}_2$ returns 0 which implies, k_0 is a valid session key and η^* is a signcryption ciphertext $(k_0 \parallel TS \parallel ID_i \parallel \psi)$, else it will return 1. The probability of selecting test session τ, when ID_A act as initiator and is S_B as responder is $\frac{1}{uv\gamma}$. Let the probability of $\mathscr{A}$ wins the game is χ. Therefore $\mathscr{C}_1$'s advantage is

$$\text{Adv}(\mathscr{C}_2) = \epsilon_2' \geq \frac{Pr[\chi \mid E]}{uv\gamma}. \quad \square$$

5.7. Security proof

The security is verified by using ProfVerif where the communication between the client and server are modeled by this tools [28]. ProVerif is a well-known cryptographic protocol verifier which is designed for the analysis of the authentication and key establishment properties. Furthermore it verifies the security properties such as privacy, traceability and verifiability.

6. Performance analysis

We evaluate the computational cost and communication over of PUA-KE and compared with the relevant protocols on authentication and key establishment protocols proposed by Fagen Li *et al.* [29], Alzubair *et al.* [30], Hsieh *et al.*,hsieh2014anonymous, Liao *et al.* [31] and Xianjiao *et al.*,zeng2018aua

These protocols are developed for Mobile Client–Server environment. We have ignored the other phases performed in these protocol and adopt some results and borrowed data from the work done by Challa *et al.* [32]. We use the following notations to denote the execution time of various cryptographic operations to evaluate the computational time.

- T_{pair}: time to compute pairing operation $\hat{e} : G_1 \times G_1 \rightarrow G_2$.
- T_{pm}: time to compute ECC point multiplication in the additive group G_1.
- T_{me} : time to compute modular exponent.
- T_h : time to compute hash function.

Algorithm 1 Type Declarations

```
free c:channel.
free cfr:channel.
(*–constants—*)
type QP.
type mkey.
type uq.
type exponent.
type host.
const q:QP.
const s:uq.
free Q,W1,W2,W3:QP.
free a1,a2,a3:bitstring[private].
free Ppub:QP.
const e1:uq.
(* The identities are assumed to be the identities of the attacker A and
Server B *) free IDA:uq.
free IDB:uq.
free findA:QP.
(*The element that needs to be found the session key of Server B*)
free cl:channel[private].
table d(host,uq).
table d1(host,QP).
table d2(host,QP).
table d3(host,QP).
(* Full Private Key.*)
```

We present an empirical analysis with respect to computational time and communication overhead at each security level. The Tables 7 and 8 illustrate the computational overhead and communication time respectively. We have compared the quantitative comparison with our protocol PUA-KE. The comparison of computational time in millisecond is presented in Fig. 4 and comparison of communication overhead at each security level in bytes is presented in Fig. 5. We adopt JPBC: Java pairing based cryptography [35] considering Type A pairing in our analysis. The following elliptic curve is chosen over the prime field. The Type-A pairing is chosen on this underline curve.

$$y^2 \equiv (x^3 + x) \bmod q$$

where $q \equiv 3 \bmod 4$. The embedding degree is two and the order of G_1 is p. The experiment is done on USRP2 software radios [36] equipped with 400 MHz daughter boards for compatibility and 402–405 MHz Medical Implant Communication Services (MICS) band used by IMDs [37]. The evaluation has been done using Medtronic Virtuoso implantable cardiac defibrillator (ICD) and the Concerto cardiac. We evaluate the computational time and communication overhead at 80-bit, 112-bit and 128-bit key size security levels. The size of the two prime p and q are specified in Table 6 [2]. In order to compute the communication overhead, we follow Table 5 which specifies the size of elements and the parameter used in the protocols. Tables 2 and 8 shows the execution time expression at client and the server end and communication overhead expression respectively. The computational time at the client and server end for Fagen Li *et al.* is calculated as $2T_{pm} + 1T_h + 2T_{me} = 2 * 0.0171 + 1 * 0.00032 + 2 * 0.0192 = 0.07292$ at client side and $2T_{pair} + 1T_{pm} + 1T_h + 2T_{me} = 2 * 0.0496 + 1 * 0.0171 + 1 * 0.00032 + 2 * 0.0192 = 0.15502$ at server side respectively and total computation time $0.07292 + 0.15502 = 0.22794$ s, for Alzubair *et al.* $T_{pair} + T_{pm} + 4T_h = 0.0496 + 0.0171 + 4 * 0.00032 = 0.06798$ and $3T_{pair} + 3T_h = 3 * 0.0496 + 3 * 0.00032 = 0.14976$ at server and client side respectively and a total time of 0.21774 s, In Hsieh *et al.* [33]'s scheme the execution time at user's end is $7T_{pm} + 7T_h$ and server's end is $2T_{pair} + 5T_{pm} + 5T_h$. So the execution time is $7 * 0.0171 + 7 * 0.00032 = 0.12194$ and $2 * 0.0496 + 5 * 0.0171 + 5 * 0.00032 = 0.1863$ at client and server end respectively. So the total is 0.30824 millisecond. In Liao

Table 2
Computation Time.

Protocols	Client	Server	Total
Fagen Li *et* al. [29]	$2T_{pm} + 1T_h + 2T_{me}$	$2T_{pair} + 1T_{pm} + 1T_h + 2T_{me}$	$3T_{pm} + 2T_h + 4T_{me} + 2T_{pair}$
Alzubair *et* al. [30]	$T_{pair} + 1T_{pm} + 4T_h$	$3T_{pair} + 3T_h$	$4T_{pair} + 1T_{pm} + 7T_h$
Hsieh *et* al. [33]	$7T_{pm} + 7T_h$	$2T_{pair} + 5T_{pm} + 5T_h$	$12T_{pm} + 2T_{pair} + 12T_h$
Liao *et* al. [31]	$7T_{pm} + 6T_h$	$2T_{pair} + 5T_{pm} + 3T_h$	$12T_{pm} + 2T_{pair} + 9T_h$
Xianjiao *et* al. [34]	$2T_{pm} + 1T_{me} + 6T_h$	$2T_{pair} + 2T_{me} + 5T_h$	$2T_{pm} + 3T_{me} + 2T_{pair} + 11T_h$
PUA-KE	$4T_{pm} + 4T_h + 1T_{me}$	$2T_{pm} + 2T_h + 1T_{me}$	$6T_{pm} + 6T_h + 2T_{me}$

Algorithm 2 Functions

```
fun pm(QP,bitstring):QP.
equation forall x:bitstring,y:bitstring;
pm(pm(q,x),y)=pm(pm(q,y),x).

fun e(QP,QP):QP.
fun exp(QP,exponent):QP.
(*Addition definition for multiple types*)

fun add(bitstring,bitstring):bitstring
fun add1(uq,bitstring):uq.
fun adds(uq,uq):uq.
fun minus(QP,bitstring) :QP.
fun mult(uq,QP):QP.
equation forall X:QP;
mult(e1,X)=X.
fun addQP(QP,QP):QP.
fun mulQP(QP,QP):QP.
fun concate(bitstring,bitstring):bitstring.
(*Type Conversions*)

fun stoQP(uq):QP[typeConverter].
fun stoBS(uq):bitstring[typeConverter].
fun utoE(uq):exponent[typeConverter].
fun QPtoBS(QP):bitstring[typeConverter].
fun ntoBS(uq):bitstring[typeConverter].
fun BStom(bitstring):mkey[typeConverter] .
fun div(uq):uq.
equation forall x:uq;
div(div(x))=x.
fun ntom(uq,uq):mkey.

(*Other Random Mathematical Operations*)

fun h(bitstring):bitstring.
fun XOR (bitstring,bitstring):bitstring.

(*Encryption Functions*)

fun mac(bitstring,mkey):bitstring.
reduc forall m: bitstring , k : mkey ;
get_message(mac(m, k)) = m.
fun enc_1(bitstring) : uq.
fun enc_4(bitstring,bitstring,QP,QP):uq.
fun enc_3(QP):bitstring.
fun enc_2(QP): uq.
```

Algorithm 3 The Registration Process.

```
new IDi:uq;
new c1:channel;
new r:uq;
new n:uq;
event register(hx,IDi);
out(c1,stoBS(IDi));
in(c1,ID1:bitstring);
out(c1,enc_1(ID1));
in(c1,s1:uq);
out(c1,adds(s1,s));
in(c1,t:uq);
out(c1,div(t));
in(c1,t:uq);
out(c1,pm(Q,stoBS(t)));
in(c1,k:QP);
insert d1(hx,k);
out(c1,mult(r,addQP(Ppub,pm(Q,stoBS(enc_1(ID1))))));
in(c1,r1:QP);
out(c1,div(adds(n,enc_2(r1))));
in(c1,skp:QP);
insert d2(hx,skp);
out(c1,mulQP(skp,k));
in(c1,skf:QP);
insert d3(hx,skf);
event registre(hx,IDi,r1,skf).
```

et al. [31]'s scheme, the cost at user's and server's end are $7T_{pm} + 6T_h = 7 * 0.0171 + 6 * 0.00032 = 0.12162$ and $2T_{pair} + 5T_{pm} + 3T_h = 2 * 0.0496 + 5 * 0.0171 + 3 * 0.00032 = 0.18566$ respectively. So the total execution time is $12T_{pm} + 2T_{pair} + 9T_h = 12 * 0.0171 + 2 * 0.0496 + 9 * 0.00032 = 0.30728$ millisecond. The execution time in Xianjiao *et al.* [34]'s scheme is $2T_{pm} + 1T_{me} + 6T_h = 2 * 0.0496 + 0.0192 + 6 * 0.0032 = 0.05532$ and $2T_{pair} + 2T_{me} + 5T_h = 2 * 0.0496 + 2 * 0.0192 + 5 * 0.0032 = 0.1392$ at user's and server's sides respectively. The total computational time is given by 0.2912 ms. In the proposed PUA-KE's scheme the execution time at user's side is $4T_{pm} + 4T_h + 1T_{me} = 4 * 0.0171 + 4 * 0.00032 + 0.0192 = 0.08888$ and at server's side, it is $2T_{pm} + 2T_h + 1T_{me} = 2 * 0.0171 + 2 * 0.00032 + 0.0192 = 0.05404$. So the total execution time is given by 0.14292 millisecond.

Table 6 illustrates the security level specification in bits and Table 5 summarizes the size of group elements and other parameters at the 80-bit, 112-bit and 128-bit security levels. We assume that $|TS| = 32$ bits and $|ED| = 112$ bits. For 80-bit security level the size of p, q, G_1 and G_2 are 160, 512, 1024 and 1024 bits, respectively. Using the compression method in [38], we can reduce the size of an element in G_1 to 65 bytes and G_2 to 128 bytes. Therefore, the communication overheads of Fagen Li *et al.* [29], Alzubair et al. [30], Hsieh *et al.* [33], Liao *et al.* [31]and the proposed PUA-KE scheme are $2|G_1| + |ID| + |H| + |TS| + |ED| + |k| = 2 * 65 + 20 + 20 + 4 + 14 + 10 = 198$ bytes, $2|G_1| + |ID| + 2|\mathbb{Z}_q^*| = 2 * 65 + 20 + 2 * 64 = 278$ bytes, $7|G_1| + 2|H| + |ED| = 7 * 65 + 2 * 20 + 14 = 509$ bytes, $5|G_1| + |ID| + 2|H| + |ED| = 5 * 65 + 4 + 2 * 20 + 14 = 399$ and $2|G_1| + |ID| + |H| + |TS| + |ED| + |k| = 2 * 65 + 20 + 20 + 4 + 14 + 10 = 198$ bytes, respectively. The Xianjiao *et al.* [34]'s protocol performed in four phases as user registration, server registration, online login and authentication and password change. We assume that the length of password is same as the length of use's identity of 32 bits. Communication overhead in each phases are given by (i) user registration $|G_1| + 2|H| + |ID|$, (ii) server registration $|ID| + |G_1|$, (iii) online login and authentication $2|G_1| + 2|H| + 256$ and (iv) password change $3|ID|$. So the total communication

Table 3
Comparison of security.

Security attributes	Fagen Li et al [29]	Alzubair et al [30]	Hsieh et al. [33]	Liao *et* al. [31]	Xianjiao *et* al. [34]	PUA-KE
Off-line dictionary Attack	No	No	No	No	Yes	Yes
Denial of Service Attack	No	No	No	No	Yes	Yes
Mutual Authentication	Yes	No	Yes	Yes	Yes	Yes
User Anonymity	Yes	No	Yes	No	Yes	Yes
Un-traceability	Yes	Yes	Yes	No	No	Yes
Perfect forward secrecy	Yes	Yes	Yes	Yes	Yes	Yes
Known Key security	No	No	Yes	No	Yes	Yes
Reply Attack	No	No	Yes	No	Yes	Yes
Key agreement	Yes	Yes	Yes	No	No	Yes

Algorithm 4 Client Side Authentication.

```
new IDi:uq;
get d(=hx,IDi) in
if(IDi=IDA)
then event continue(hx)
else event check(hx);
new k :bitstring;
new r:exponent;
new TS:bitstring;
new c2:channel;
new n:uq;
out(c2,exp(W1,r));
in (c2,alpha:QP);
out (c2,enc3(alpha));
in (c2,h1:bitstring);
get d(=hx,ID1) in
out (c2,XOR(concate(concate(k,TS),stoBS(ID1)),h1));
in (c2,C:bitstring);
out (c2,enc4(C,C,alpha,alpha));
in (c2,h0:uq);
get d3(=hx,sk1) in
out (c2,mult(adds(n,h0),sk1));
in (c2,V:QP);
get d(=hy,ID1) in
get d2(=hx,pk1) in
get d2(=hy,pk2) in
get d(=hy,ID2) in
out (c2,mult(n,addQP(pk1,mult(enc2(pk2),
addQP(mult(enc1(stoBS(ID2)),Q),Ppub)))));
in (c2,Z:QP);
out (c,(C,V,Z,k));
event fullC(hx);
in(cl,(b1:bitstring,k1:bitstring));
out(c2,mac(XOR(k,k1),BStom(TS)));
in(c2,b2:bitstring);
if(b1=b2)
then event forgery(b1).
```

Algorithm 5 Server Side Authentication

```
get d(=hx,IDi) in
if(IDi=IDB)
then event continue(hx)
else event check(hx);
get d2(=hx,findA) in
event partialS(hx);
in(c,(C:bitstring,V:QP,Z:QP,k:bitstring));
new c3:channel;
get d3(=hy,skj) in
out(c3,e(Z,skj));
in(c3,alpha:QP);
out(c3,enc3(alpha));
in(c3,h1:bitstring);
out(c3,XOR(C,h1));
in(c3,Ch1:bitstring);
out(c3,enc4(Ch1,Ch1,alpha,alpha));
in(c3,h:uq);
get d(=hx,ID1) in
get d2(=hx,pk1) in
get d(=hy,ID2) in
out(c3,e(V,addQP(addQP(pk1,mult(enc2(pk1),
mult(enc1(stoBS(ID2)),Q))),Ppub)));
in(c3,alpha:QP);
out(c3,mulQP(alpha,exp(W1,utoE(h))));
in(c3,alpha:QP);
out(c3,mac(XOR(k,k1),BStom(TS)));
in (c3,b1:bitstring);
out(cl,XOR(k,k1));
in(cl,r1:bitstring);
out(cl,(b1,k1 )).
```

Table 4
Computation overhead.

Protocols	Communication Overhead
Fagen Li et al. [29]	$2\|G_1\| + \|ID\| + \|H\| + \|TS\| + \|ED\| + \|k\|$
Alzubair et al. [30]	$2\|G_1\| + \|ID\| + 2\|\mathbb{Z}_q^*\|$
Hsieh et al. [33]	$7\|G_1\| + 2\|H\| + \|ED\|$
Liao et al. [31]	$5\|G_1\| + \|ID\| + 2\|H\| + \|ED\|$
Xianjiao *et* al. [34]	$4\|G_1\| + 4\|H\| + 5\|ID\| + 256$
PUA-KE	$2\|G_1\| + \|ID\| + \|H\| + \|TS\| + \|ED\| + \|k\|$

overhead is given by $4|G_1|+4|H|+5|ID|+256 = 4*65+4*20+5*4+32 = 392$ bytes. Similarly, we can obtain the communication overheads at the 112-bit and 128-bit security levels. These are presented in Table 4 at 80-bit, 112-bit and 128-bit security level. The Table 3 depicts the security comparison of

7. Conclusion

The paper presents an efficient and lightweight user authentication and key establishment protocol for IMD. Our scheme is based on CL-PKC and the security is proven in random oracle model. In order to allow the user to access the data stored in CD, the user needs a successful authentication. Our protocol achieves authentication and establish a secure communication. We have evaluated the computational cost in term of computational time and communication overhead. Further we have computed the communication overhead at 80-bit, 112-bit and 128-bit security level. We observed that, the communication overhead is relatively less that the other such protocols. We have proven that the proposed scheme (PUA-KE) provides a secure key establishment protocol.

Algorithm 6 Events and queries

```
(* Queries *)

event register(host,uq).
event registre(host,uq,QP,QP).
event partialS(host).
event fullC(host).
event continue(host).
event forgery(bitstring).
event check(host).
event cr(uq,uq).

axiom ID1:uq,ID2:uq,ID3:uq;
event(cr(ID1,ID2)) ==> ID1<>ID2.
axiom hx:host,ID1:uq,ID2:uq,pk1:QP,pk2:QP,sk1:QP,sk2:QP;
event(register(hx,ID1)) &&
event(register(hx,ID2)) ==> (ID1=ID2)[induction].
restriction hr:host;
event(partialS(hr)) ==> event(check(hr)).
query x:bitstring;event(forgery(x))[induction].
```

Table 5
Size of group elements and other parameter in Byte.

Size of elements /parameters	80-bit	112-bit	128-bit
G_1	65	130	260
G_2	128	256	384
H	20	28	32
ID	20	28	32
k	10	14	16
TS	4	6	8
ED	14	18	22

Table 6
Security level specification in bits.

Security level	Size of p	Size of q
80-bit	512	160
112-bit	1024	224
128-bit	1536	256

Table 7
Communication Overhead (Byte)

Protocols	80-bit	112-bit	128-bit
Fagen Li *et* al. [29],	198	354	632
Alzubair *et* al. [30]	278	544	936
Hsieh *et* al. [33]	509	984	1902
Liao *et* al. [31]	383	752	1418
Xianjiao *et* al. [34]	392	1028	1584
PUA-KE	198	354	632

Table 8
Computational time in millisecond.

Protocol	Client	Server	Total
Fagen Li *et* al. [29]	0.07292	0.15502	0.22794
Alzubair *et* al. [30]	0.06798	0.14976	0.21774
Hsieh *et* al. [33]	0.12162	0.18566	0.30728
Liao *et* al. [31]	0.34912	0.09542	0.44454
Xianjiao *et* al. [34]	0.05532	0.1392	0.2912
PUA-KE	0.08888	0.05404	0.14292

Declaration of competing interest

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

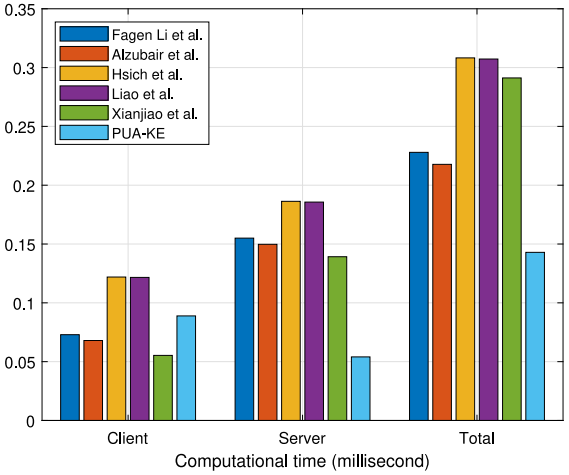

Fig. 4. The computational time in millisecond.

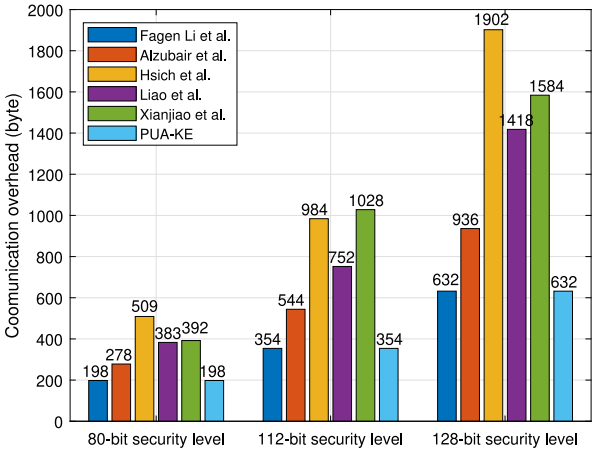

Fig. 5. The communication overhead at each security level.

References

[1] A. Alsuwaidi, A. Hassan, F. Alkhatri, H. Ali, M. Qbea'H, S. Alrabaee, Security vulnerabilities detected in medical devices, in: 2020 12th Annual Undergraduate Research Conference on Applied Computing (URC), 2020, pp. 1–6, http://dx.doi.org/10.1109/URC49805.2020.9099192.

[2] J. Kar, K. Naik, T. Abdelkader, An efficient and lightweight deniably authenticated encryption scheme for e-mail security, IEEE Access 7 (2019) 184207–184220.

[3] D. He, S. Zeadally, Authentication protocol for an ambient assisted living system, IEEE Commun. Mag. 53 (1) (2015) 71–77.

[4] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, K. Fu, They can hear your heartbeats: non-invasive security for implantable medical devices, in: Proceedings of the ACM SIGCOMM 2011 Conference, 2011, pp. 2–13.

[5] D. He, S. Zeadally, N. Kumar, J.-H. Lee, Anonymous authentication for wireless body area networks with provable security, IEEE Syst. J. 11 (4) (2016) 2590–2601.

[6] X. Li, J. Niu, S. Kumari, F. Wu, K.-K.R. Choo, A robust biometrics based three-factor authentication scheme for global mobility networks in smart city, Future Gener. Comput. Syst. 83 (2018) 607–618.

[7] W. Wang, H. Huang, L. Xue, Q. Li, R. Malekian, Y. Zhang, Blockchain-assisted handover authentication for intelligent telehealth in multi-server edge computing environment, J. Syst. Archit. 115 (2021) 102024.

[8] Z. Gao, Z. Cheng, W. Diao, J. Zhang, H. Lu, Identity authentication based on trajectory characteristics of mobile devices, J. Syst. Archit. 112 (2021) 101857.

[9] 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, Multimedia Syst. 21 (1) (2015) 49–60.

[10] X. Hei, X. Du, Biometric-based two-level secure access control for implantable medical devices during emergencies, in: 2011 Proceedings IEEE INFOCOM, IEEE, 2011, pp. 346–350.

[11] F. Xu, Z. Qin, C.C. Tan, B. Wang, Q. Li, IMDGuard: Securing implantable medical devices with the external wearable guardian, in: 2011 Proceedings IEEE INFOCOM, IEEE, 2011, pp. 1862–1870.
[12] K.B. Rasmussen, C. Castelluccia, T.S. Heydt-Benjamin, S. Capkun, Proximity-based access control for implantable medical devices, in: Proceedings of the 16th ACM Conference on Computer and Communications Security, 2009, pp. 410–419.
[13] C. Jang, D.G. Lee, J.-W. Han, J. Park, Hybrid security protocol for wireless body area networks, Wirel. Commun. Mob. Comput. 11 (2011) 277–288, http://dx.doi.org/10.1002/wcm.884.
[14] N. Ravanbakhsh, M. Nazari, An efficient improvement remote user mutual authentication and session key agreement scheme for E-health care systems, Multimedia Tools Appl. 77 (1) (2018) 55–88.
[15] A. Ostad-Sharif, D. Abbasinezhad-Mood, M. Nikooghadam, An enhanced anonymous and unlinkable user authentication and key agreement protocol for TMIS by utilization of ECC, Int. J. Commun. Syst. 32 (5) (2019) e3913.
[16] V. Sureshkumar, R. Amin, M.S. Obaidat, I. Karthikeyan, An enhanced mutual authentication and key establishment protocol for TMIS using chaotic map, J. Inf. Secur. Appl. 53 (2020) 102539.
[17] G. Mwitende, Y. Ye, I. Ali, F. Li, Certificateless authenticated key agreement for blockchain-based WBANs, J. Syst. Archit. 110 (2020) 101777.
[18] M.J. Hossain, C. Xu, C. Li, S.H. Mahmud, X. Zhang, W. Li, ICAS: Two-factor identity-concealed authentication scheme for remote-servers, J. Syst. Archit. 117 (2021) 102077.
[19] D. He, S. Zeadally, An analysis of RFID authentication schemes for internet of things in healthcare environment using elliptic curve cryptography, IEEE Internet Things J. 2 (1) (2014) 72–83.
[20] J. Kar, Provably secure certificateless deniable authenticated encryption scheme, J. Inf. Secur. Appl. 54 (2020) 102581.
[21] M. Darji, B. Trivedi, IMD-IDS a specification based intrusion detection system for wireless IMDs, Int. J. Appl. Inf. Syst. 5 (2013) 19–23, http://dx.doi.org/10.5120/ijais13-450926.
[22] M.A. Siddiqi, C. Doerr, C. Strydis, Imdfence: Architecting a secure protocol for implantable medical devices, IEEE Access 8 (2020) 147948–147964, http://dx.doi.org/10.1109/ACCESS.2020.3015686.
[23] M.A. Siddiqi, R.H. Beurskens, P. Kruizinga, C.I. De Zeeuw, C. Strydis, Securing implantable medical devices using ultrasound waves, IEEE Access 9 (2021) 80170–80182, http://dx.doi.org/10.1109/ACCESS.2021.3083576.
[24] Y. Zheng, Digital signcryption or how to achieve cost (signature & encryption) ≪ cost (signature)+ cost (encryption), in: Annual International Cryptology Conference, Springer, 1997, pp. 165–179.
[25] M. Bellare, P. Rogaway, Entity authentication and key distribution, in: Annual International Cryptology Conference, Springer, 1993, pp. 232–249.
[26] N. McCullagh, P.S. Barreto, A new two-party identity-based authenticated key agreement, in: Cryptographers' Track At the RSA Conference, Springer, 2005, pp. 262–274.
[27] M.C. Gorantla, C. Boyd, J.M.G. Nieto, On the connection between signcryption and one-pass key establishment, in: IMA International Conference on Cryptography and Coding, Springer, 2007, pp. 277–301.
[28] B. Blanchet, M. Abadi, C. Fournet, Automated verification of selected equivalences for security protocols, J. Log. Algebr. Program. 75 (1) (2008) 3–51.
[29] F. Li, J. Wang, Y. Zhou, C. Jin, S.H. Islam, A heterogeneous user authentication and key establishment for mobile client–server environment, Wirel. Netw. (2018) 1–12.
[30] A. Hassan, A.A. Omala, M. Ali, C. Jin, F. Li, Identity-based user authenticated key agreement protocol for multi-server environment with anonymity, Mob. Netw. Appl. 24 (3) (2019) 890–902.
[31] Y.-P. Liao, C.-M. Hsiao, A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients, Future Gener. Comput. Syst. 29 (3) (2013) 886–900.
[32] S. Challa, M. Wazid, A.K. Das, M.K. Khan, Authentication protocols for implantable medical devices: taxonomy, analysis and future directions, IEEE Consum. Electron. Magaz. 7 (1) (2017) 57–65.
[33] W.-B. Hsieh, J.-S. Leu, An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures, J. Supercomput. 70 (1) (2014) 133–148.
[34] X. Zeng, G. Xu, X. Zheng, Y. Xiang, W. Zhou, E-AUA: An efficient anonymous user authentication protocol for mobile IoT, IEEE Internet Things J. 6 (2) (2018) 1506–1519.
[35] B. Lynn, et al., PBC library, 2006, Online: http://crypto.stanford.edu/pbc.
[36] M. Ettus, M. Braun, The universal software radio peripheral (usrp) family of low-cost sdrs, in: Opportunistic Spectrum Sharing and White Space Access: The Practical Reality, Wiley, 2015, pp. 3–23.
[37] P.D. Bradley, An ultra low power, high performance medical implant communication system (MICS) transceiver for implantable devices, in: 2006 IEEE Biomedical Circuits and Systems Conference, IEEE, 2006, pp. 158–161.
[38] H. Zhong, B. Huang, J. Cui, J. Li, K. Sha, Efficient conditional privacy-preserving authentication scheme using revocation messages for vanet, in: 2018 27th International Conference on Computer Communication and Networks (ICCCN), 2018, pp. 1–8.

Neha Kumari is a Doctoral research scholar at Department of Computer Science and Engineering, LNM Institute of Information Technology, Jaipur Rajasthan. She obtained B.Tech. in CSE (2015), M.Tech. in Cyber Security (2017) and qualified UGC-NET (2018). She is an active member of Centre for Cryptography, Cyber Security and Digital forensics, The LNMIIT, Jaipur. She has attended many national and international conferences and workshop on Blokchain Technology and Cyber Security.

Jayaprakash Kar has received his M.Sc and M.Phil in Mathematics from Sambalpur University, M.Tech and Ph.D in Computer Science (Cryptographic Protocols) from Utkal University, India. He is visiting researcher of Department of Electrical & Computer Engineering, University of Waterloo, Canada. Currently he is working as Associate Professor in the Department of Computer Science & Engineering, The LNM Institute of Information Technology, Jaipur, India. He is Center-Lead of "Center for Cryptography, Cyber Security and Digital forensics" (C3-SDF). His current research interests are on Cryptographic protocols and primitives using Elliptic Curve and Pairing based Cryptography in Random Oracle and Standard model. Dr. Kar is advisory and editorial board member of many peer reviewed Journals and international conferences. He is Associate editor of Journal of Circuits, Systems and Computers, World Scientific. He is life member of International Association for Cryptology Research (IACR), Cryptology Research Society of India, IEEE senior member, and associate member of ACM, International Association of Computer Science & Information Technology (Singapore) and International Association of Engineers (United Sates).

Kshirasagar Naik is a Full Professor in the Department of Electrical and Computer Engineering at the University of Waterloo. Previously, he held faculty positions at Carleton University in Ottawa and University of Aizu in Japan. He worked as a software developer for three years in Wipro, Bangalore – now one of the largest software consultancy companies in the world. His research interests include vehicular networks, delay tolerant networks, energy performance testing of mobile applications, detection of anomalous behavior of wireless devices and physical systems, energy harvesting IoT (Internet of Things) devices for sustainable monitoring of physical systems, communication security, and communication protocols for smart power grids. Designing mathematical models and building prototype sensor networks for performing real-life, controlled experiments lie at the core of his research. He has served on the editorial boards of many journals, including: Journal of Peer-to-Peer Networking and Applications, International Journal of Parallel, Emergent and Distributed Systems, Journal of Circuits, Systems, and Computers, and IEEE Transactions on Parallel and Distributed Systems. He was a co-guest editor of four special issues of IEEE Journal on Selected Areas in Communications and IEEE Transactions on Cloud Computing. He is a co-author of two widely used textbooks, namely, Software Testing and Quality Assurance: Theory and Practice (Wiley, 2008) and Software Evolution and Maintenance: A Practitioner's Approach (Wiley, 2014).