Research article

Multifactor authentication scheme using physically unclonable functions

Moneer Fakroon\(^a\), Fayez Gebali\(^a\), Mohammad Mamun\(^b\)

\(^a\) Victoria University, Canada
\(^b\) National Research Council of Canada, Government of Canada, Fredericton, NB, Canada

**Abstract**

We propose a secure telehealth system using multifactor authentication for the mobile devices as well as the IoT edge devices in the system. These two types of devices constitute the weakest link in telehealth systems. The mobile devices and edge devices are typically unsecured and contain vulnerable processors. The mobile devices use the healthcare professional's biometric and endowing the edge device with biometrics is accomplished by using physically unclonable functions (PUFs). The embedded PUF acts as a means of enabling mutual authentication and key exchange. Evaluating the security of the proposed authentication scheme is conducted using three approaches: (a) formal analysis based on Burrows–Abadi–Needham logic (BAN); (b) informal security analysis for protection against many attack types; (c) model checking using automated validation of internet security protocols and applications (AVISPA) tool.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

We are witnessing the confluence of many factors that promise to impact the way our society operates: the rapid introduction of 5G and Wi-Fi 6 communication technologies, the use of artificial intelligence (AI) and machine learning (ML) and the implementation of Internet-of-Things (IoT) in industry and at home [1–3]. IoT systems now control valuable infrastructures and security of these systems against attacks becomes mandatory.

Many factors contributed to the incorporation of IoT networks for telehealth systems. In countries like Canada it is possible to deliver a higher level of health services to remote communities by offering online patient assaying/diagnosis using remote medical devices such as in body area networks. It has also been shown that providing telehealth to at-home patient care leads to faster recovery and reduces the health services costs [4,5]. People mobility between different cities, provinces or even countries suggests adoption of telehealth systems. It would be very wasteful to create new medical files each time a person moves to another province/state or country. Lately, the COVID-19 pandemic necessitated reduced face-to-face patient-healthcare professional interactions [6,7] especially for walk-in clinics and hospital emergency admission for walk-in patients.

Securing these services and infrastructure is a must given the increasing threats on IoT systems in general and on telehealth systems in particular [8–11]. The telehealth system is an essential service and has in its possession valuable informa-
tion associated with patients and data associated with hospital operations, management, and accounting. All these factors present valuable targets for cyberattacks.

The authors in [12] focused on reviewing data security using blockchain in the IoT for telehealth. These authors summarized possible IoT attacks in general as physical attacks, network attacks, software attacks and encryption attacks. The authors argue that telehealth is at the top of digital technologies that are at risk from cyber attacks. The authors stated that medical data is attacked where it is stored or when it is being transferred from one location to another.

Islam et al. [13] provided a survey of IoT for telehealth including proposed architectures for efficient telehealth delivery. Telehealth implies delivery of telehealth to stay-at-home patients through the internet cloud and remotely-located IoT networks. The services provided by telehealth include:

1. Securing IoT environment against active and passive attacks
2. Monitoring of patient through the wearable devices (posture, activity, biometrics, diabetes, ECG, etc.)
3. Secure delivery and accounting of patient medications
4. Inventory of medications, devices and sensors
5. : 

There are well established aspects of security including [14,15]

1. Authenticating system users to confirm identity and accountability.
2. Ensuring availability of the services provided.
3. Assuring confidentiality/privacy of information and accessibility by authorized users.
4. Preserving data integrity against storage/transmission due to errors or maliciously.
5. Adding non-repudiation so sender can not deny sending the message and receiver can not deny receiving the message.

In this work we focus on the authentication aspect of security. The main difficulty of delivering all the above security features to telehealth is the diversity of entities and the widespread distribution of the devices. For example, the hospital can be thought of as the service provider and can be set up with many layered security safeguards. On the other hand the IoT edge devices, like sensors and actuators, are very vulnerable since they are placed at homes where security levels cannot be guaranteed. An edge device is prone to stealing secret keys, denial of service, tampering, etc. The IoT edge devices for telehealth control the overall security of the system since they are effectively its weakest link. Table 1 summarizes the security challenges of IoT for telehealth systems.

Several authors proposed authentication schemes which allow patients to remotely access telehealth and consult with healthcare professionals (doctors or nurses) [16]. The schemes proposed by these authors identified five main players: patients, healthcare professional, body sensors in some patients, telehealth centre and cloud server. The proposed schemes are claimed to ensure mutual authentication, resistance to impersonation attacks, resistance to replay attacks, message authentication, forward/backward secrecy of the session key, non-repudiation, anonymity and unlinkability, and confidentiality.

Chen et al. [17] proposed a systematic analysis of IoT remote binding between the IoT users and IoT devices through the cloud. The authors divided remote binding into four phases: (1) Authenticating the user and device to the cloud; (2) Binding between user and device on the local network; (3) User controls device by providing device-specific secret to the cloud as credential; (4) Revoke binding at end of session. These functionalities were then described using state-machine model. It was claimed that the state-machine model helps to systematically analyze the binding operation and measure the security vulnerabilities.

Yu and Li [18] proposed an anonymous authentication key agreement scheme for multi-sensor home-based IoT. The proposed scheme used lightweight authentication and key agreement technology using pairing-based cryptography. A lightweight scheme was used due to the limited communication and processing capabilities of the edge devices.

Wazid et al. [19] proposed a secure mobile device authenticated key establishment for smart home environment using only simple one-way hashing, bitwise XOR operations and symmetric key encryption. However, the scheme used verification tables that are vulnerable since they are stored in the IoT gateway. The protocol used time stamps thus making it vulnerable to synchronization attacks.

In an earlier publication [20], Fakroon et al. proposed a lightweight and secure user mutual authentication and key agreement protocol for smart home access. The protocol avoided clock synchronization problem that exists in some time-

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Security challenges of IoT for telehealth systems.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>Comment</td>
</tr>
<tr>
<td>Architecture</td>
<td>Massively heterogenous hardware, software and protocols</td>
</tr>
<tr>
<td>Diversity</td>
<td>Diverse devices and capabilities. Servers are state-of-the-art and dated edge devices</td>
</tr>
<tr>
<td>Defences</td>
<td>Layered starting from hardware to software applications</td>
</tr>
<tr>
<td>Protection</td>
<td>Not all entities can be protected, especially the edge devices</td>
</tr>
<tr>
<td>Encryption</td>
<td>Should be hardware-based encryption, hashing and true random number generation</td>
</tr>
<tr>
<td>Threats</td>
<td>Multiplying and changing daily</td>
</tr>
<tr>
<td>Updates</td>
<td>Protected debugging/programming ports for edge devices</td>
</tr>
<tr>
<td>Vulnerabilities</td>
<td>Servers can be hardware root-of-trust (HRoT) while edge devices are very vulnerable</td>
</tr>
</tbody>
</table>
stamped authentication protocols. The protocol proposed by Fakroon et al. differed from the protocol proposed here in several aspects:

1. The number of entities considered by Fakroon et al. [20] is three: User, Gateway, and Smart device. On the other hand, our proposed considers four entities: Healthcare professional, Server, Gateway, and IoT Edge device.
2. Fakroon et al. allowed the User, which is equivalent to our Healthcare professional, to communicate directly with the IoT gateway. In our case we allowed the Healthcare professional to communicate only with the Server. This has the advantage of confronting possible attacks by the server and gateway as opposed to relying on the mobile device and edge device with their limited capabilities.
3. Fakroon et al. used the User’s physical location for context awareness as a means to establish authentication. In our work we used context awareness of the mobile devices based on its location and context awareness of the IoT edge devices and the context is based on identifying the other neighbouring IoT edge devices.
4. Fakroon et al. authentication and secret key generation of IoT edge devices was based on a unique secret key defined by the manufacturer and implicitly stored in non-volatile memory. In our case authentication and secret key generation are based on using physically unclonable functions PUF, as explained in Section 3.1. This improved security of the system since edge devices now have biometrics and secret key is not stored in vulnerable non-volatile memories.

Chen et al. [21,22] proposed a new scheme for patient’s privacy based on the cloud computing. Mobile device characteristics were used to allow people to use medical resources on the cloud environment. The scheme did not support patient anonymity or message authentication.

Chiou et al. [23] improved on the scheme proposed by Chen et al. [21] to reduce computation costs while achieving patient anonymity and unlinkability, and message authentication.

Mohit et al. [24] proposed a mutual authentication protocol for cloud computing based telehealth system. However Li et al. [25] found design flaws in the proposed protocol in that it did not provide security against health report revelation, inspection report forgery and patient anonymity and unlinkability.

Contributions: The contributions of this article are as follows:

1. Propose a secure lightweight mutual authentication and key exchange scheme. The proposed scheme is lightweight since it only uses simple operations to implement the hash functions and XOR operations. Other protocols used very complex finite field operations typically used in complex cryptographic operations such as elliptic curves. The scheme does not use a verification table for authentication and avoids the clock synchronization problem.
2. Identify and consider the different authentication requirements of the main entities of the telehealth system: server (hospital), healthcare professionals, IoT gateway and IoT edge devices.
3. Use hash chains extensively for authenticating all the telehealth entities.
4. Evaluate the security of the proposed scheme is conducted using formal analysis BAN logic, informal security analysis, and model checking using AVISPA tool.
5. Compare our proposed scheme with other related protocols and comparison results are presented in terms of security and performance.

Organization: The rest of the paper is organized as follows. Section 2 summarizes the notations used in this work. Section 3 discusses silicon physically unclonable functions (PUF), threat model, and models for the main players in the healthcare system. Section 4 describes the proposed mutual authentication scheme and its main phases of operation. Section 5 provides security analysis of the proposed scheme including formal proof using BAN logic, informal security analysis, and performance comparison.

2. Notation and terms used

Table 2 summarizes the notations used in this work.

3. Preliminaries

In this section we describe the models and assumptions related to the agents or entities interacting with the IoT telehealth system. The main actors in this system include the threat/adversary, the edge devices and the users.

3.1. Silicon physically unclonable function (PUF)

The use of physically unclonable functions (PUF) for mutual authentication in IoT devices has been the recognized solution to endowing IoT devices with a unique identity, akin to a fingerprint or retina image for humans. A PUF serves to authenticate a device and also provides a measure of tamper resistance.

Delvaux et al. [26,27] reviewed designs of strong and weak PUFs and how they can be used in entity authentication. The authors provided a review of the two types of PUF circuits and their statistical properties. Delvaux also provided an excellent survey of PUF-based key generation. Discussion was provided on how the helper data algorithm can be used to extract a stable secret key from the noisy PUF response (see Chapter 4 in [27]).
Dodis et al. [28,29] provided efficient secure techniques for using biometrics to provide secure device authentication and converting the noisy biometric data into stable cryptographic keys. They indicated that any biometric signal inherently has low entropy and is also noisy. On the other hand, any useful cryptographic key must have high entropy and be stable and noise free. For this end, the technique they introduced depends on a fuzzy extractor or helper data algorithm. Fig. 3 in the sequel shows the main building blocks of the secure sketch or fuzzy extractor. Fig. 4 in the sequel shows only the secure key generation part of the fuzzy extractor.

Ravikanth [30,31] discussed the concept of one-way functions in general and distinguished between the similarities between algorithmic one-way functions (e.g. RSA and Rabin functions) versus physical one-way functions (e.g. PUFs) which depend on physical phenomena for their operation. He discussed the operation of optical one-way functions and reported the like and unlike binomial distribution of the resulting optical responses to ensure the uniqueness of the optical fingerprint of the devices.

Gassend et al. [32] discussed the more common silicon physical random functions, which are now commonly known as physically unclonable functions (PUF). They studied different types of delay-based PUF such as self-oscillating circuits exhibiting monotonic and non-monotonic delays. They also discussed generating redundant information to be added to the challenge-response pairs (CRP) as a means of error correction to remove the noise from the responses and implemented the designs on FPGA hardware.

Suh et al. [33] discussed two different types of delay-based silicon PUFs: ring oscillator PUF and arbiter PUF. Techniques to generate sufficiently large number of CRP by increasing the options for configuring or selecting the circuit delays. A low-cost authentication scheme is proposed that does not use resource-hungry cryptographic techniques. Error detection and correction techniques using BCH coding/decoding are used for reliable cryptographic key generation based on the CRP used. The authenticator generates the redundant information and sends it in the clear to the device to be authenticated to remove the random process variations.

Guajardo et al. [34] proposed PUF SRAM structures suitable for today’s FPGA technology. The statistical properties of the SRAM were investigated. The fuzzy extractor or helper data algorithm was used to extract one or more secure keys. The authenticator generates the helper data W based on the golden PUF response that does not include thermal noise and sends this to the device. Referring to Fig. 3 and Fig. 4 in the sequel, the device to be authenticated receives the helper data r and uses it, together with the noise PUF response (w'), to generate a stable and secure session secret key.

Maes et al. [35–37] reviewed electronic and non-electronic PUF types and discussed the Hamming inter-distance and intra-distance as a means of ensuring unique device ID and to be able to distinguish between the different devices. Seven different types of PUFs were reviewed, PUF-based key generation properties were identified and the secure sketch, strong and fuzzy extractor technique proposed by Dodis et al. [28,29] were discussed.
Herder et al. [38] described using PUF for a low-cost authentication and key generation applications. The paper defined two primary PUF types: “strong PUFs” and “weak PUFs”.

Operation of the PUF relies on a challenge-response pair (CRP) where the server issues a challenge and the IoT device, or client, provides a response that is unique to the device. The problem with PUF response is it is noisy but has low entropy. Therefore techniques have been developed to recover a reliable and stable response from the noisy responses using fuzzy extractor or secure sketch [28,29,39,40]. The advantage of the fuzzy extractor serves also to generate a secret key with high entropy from the low-entropy noisy responses. Silicon Physically unclonable functions (PUF) are circuits that make use of inevitable random process variations (RPV) and thermal noise present in integrated-circuits (IC). A PUF operates on a challenge-response pair (CRP) where each challenge or input to the PUF generates a response that is unique to each manufactured IC and can not be repeated by an attacker or even the IC manufacturer. The function produced by the PUF should be easy to evaluate but hard to model or characterize by the designer, manufacturer or even the adversary. Examples of PUFs include [33] arbiter PUFs, ring oscillator PUFs and SRAM PUFs [41].

A black-box model of a silicon PUF is represented as

$$R[1 : M] = PUF(C[1 : N])$$

where $C[1 : N]$ is the $N$-bit challenge, $R[1 : M]$ is the $M$-bit response, and $PUF(\cdot)$ is the one-way unique function characterizing the PUF given the physical parameters of a particular IC.

The simple CRP set is not practical for authentication for the following reasons:

1. The responses are noisy since they are subject to processing variations and environmental factors such as temperature, supply voltage variations and ground bounce [42].
2. A CRP can only be used once to prevent replay attack. Hence, a finite number of authentication operations are feasible after which the device is withdrawn from service.
3. There is nonzero probabilities of false rejection and false acceptance.
4. Publicly exchanging the CRP on an open and un-secure channel and to an un-secure edge device significantly reduces the unpredictability of the remaining CRPs. An attacker using machine learning algorithms can quickly train for responding to a new challenge [27,43,44].

### 3.1.1. PUF-based secret key generation and storage

Countering random manufacturing variations, ambient effects, aging, and thermal noise is feasible using helper data algorithm [34,39,43] or fuzzy extractors [28,29,36]. These approaches allow a PUF to resist forgery and duplication while generating device-unique keys that are not visible when the device is powered off. Avoiding use of nonvolatile memory to store secret keys prevents theft of the cryptographic keys.

However, associating a challenge response pair for each device is possible only after the device is manufactured since no accurate model is possible to describe the PUF of a given IC. Authenticating a PUF-based device and establishing a secret key are done by the manufacturer for each device fabricated. This is because the fingerprint of each device must be obtained by explicitly examining each device. Another reason is that it is impossible to create an equivalent model for the PUF associated with a specific device. The procedure for generating and regenerating the session key are described in Sections 3.7.1.

### 3.1.2. PUF-based device enrollment

Enrolment of a PUF-based device is done by the manufacturer where each device is assigned a unique ID and a set of challenge-response pairs (CRP) and a secret key associated with each CRP. In addition, the manufacturer generates publicly available helper data for each CRP so that the device can use the noisy response together with the helper data to generate the same secret key to be used to help authenticate the device, prove freshness of the session and encrypt/decrypt data [27,46].

### 3.2. Threat model

The adversary launches targeted attacks through either the edge device or the mobile device or by intercepting the communication across the cloud. The weak link in the IoT telehealth is the edge devices since they usually have limited compute resources and seldom undergo a software/firmware updates. The Dolev–Yao model for the adversary is assumed. In addition, the adversary can

1. Infer the device ID through brute force or guessing based on knowledge of the device brand and ID sequence assignment.
2. Gain physical access to the edge devices in the field and extract stored information.
3. Launch various attacks to steal the device secret keys through reading the flash or solid-state drive (SSD) content through fault injection, memory permanence or cold boot attacks for example.
4. Launch a passive attack using side-channel analysis on the edge devices.
5. Change the flash or SSD content to run malicious software.
3.3. Network model

There communication network comprising the IoT-based healthcare system has the following main entities:

1. Internet cloud which could rely on 5G and Wi-Fi 6 technologies for increased throughput and reduced latency.
2. Secure Server S connecting the mobile devices to the internet. The server could be considered a hardware root-of-trust (HRoT) since it contains tamper-proof security processors and implements layered security protocols.
3. Gateway G connecting the edge devices to the internet. The gateway could be considered a hardware root-of-trust (HRoT) since there is one gateway in each remote location so it would not impose too much expense on the system deployment in relation to the security benefits dividends. G contains tamper-proof security processors and implements layered security protocols also.
4. Mobile devices M which are used by the healthcare professionals to access telehealth system through the server. The mobile devices are located where healthcare professionals practice their profession at clinics, hospitals, research centres, etc. The mobile devices could be connected to the server through secure virtual private networks (VPN) to reduce the number of possible attacks.
5. IoT edge devices D which are located at the local IoT network, which are typically located at the patient’s home, or remote healthcare centre, and are connected through a local-area network. These devices typically have limited compute capabilities and small memory footprint. However, it is now not too expensive to include PUF circuitry to help authenticate these devices and securely generate secret keys.

Some authors added an extra entity to the above components—mainly “patients without sensors”. We do not include such patients as a member of the IoT system since interaction between those patients and healthcare professionals is done using separate channels such as email, phone, virtual meetings, etc. The IoT system is in charge of the hardware, sensors, and actuators in the patients’ home or the body area network, if it exists.

3.4. Client model

Fig. 1 shows the client-side architecture of the IoT for the telehealth system under consideration.

The client-side IoT system represents the infrastructure at the patient’s home or the remote health care delivery site. The system consists of a gateway that provides access to the Internet through its communication layers. In addition the gateway maintains the local IoT security by serving as a local hardware-based root-of-trust (HRoT). This is not a severe requirement since price of a trusted platform module (TPM), or crypto processor, is getting lower and lower and the benefits it provides far exceeds the cost [47]. In addition to the gateway, the client-side IoT contains n edge devices \( E_1, E_2, \cdots, E_n \) that serve as sensors and/or actuators such as insulin pump, pacemaker, etc. In that sense, the gateway is the first line of defence against local attacks emanating from malicious or compromised devices. It also protects the system from remote attacks arriving through the Internet cloud. Authentication of the devices is the responsibility of the gateway. The gateway requires assistance naturally from a registration authority (RA) as well as the telehealth server discussed in the paragraph below.

3.5. Server model

Fig. 2 shows the server-side architecture of the telehealth system under consideration.

The server is located at the telehealth system infrastructure at the hospital or an equivalent infrastructure. It is supported by information technology (IT) personnel and security experts to maintain layered security measures. The computing resources of the server can be assumed limitless. Security is maintained at the application level down to the hardware level. A hardware root-of-trust (HRoT) must be present in order to secure the cryptographic keys and cryptographic primitives and protocols.
3.6. Mobile device model

Fig. 2 shows the mobile devices of the IoT telehealth system which helps access by the healthcare professionals. In addition, the mobile device could be replaced with an app that runs at the remote telehealth authority and connects to a certain IoT to poll the data of all edge devices and check the status of the actuators. This app could be used later on using machine learning (ML) techniques to infer the behaviour of the IoT and its connected devices.

The mobile devices are vulnerable to several types of attacks such as phishing, theft, reverse engineering, etc.
3.7. Edge device model

The IoT edge device used for remote telehealth is an embedded system where secret keys used for cryptographic primitives are PUF-based as was discussed in Section 3.1.1. Using a PUF ensures each device has a unique ID and a session secret key that is generated by the hardware and not stored in vulnerable nonvolatile memory [32,33,36,37,48].

Authentication of a device is essentially based on a challenge-response pair (CRP). The server issues a challenge c and generates the expected response w. The client is the edge device that receives c and generates a noisy response w'. Dodis et al. and others [27,28,49,50] provided a description of how the noisy data of a PUF can be used to generate a consistent session key to be used for authentication and secure message exchange.

3.7.1. Fuzzy extractor

Fig. 3 shows the basic structure of the fuzzy extractor technique for extracting helper data from the PUF response at the server (left) and client or IoT device (right) [27].

The server selects a challenge c and uses the dataset supplied by the manufacturer to extract the expected response w. The server also performs forward error correction coding (FEC) on the response to produce helper data r, session key K and a hash value h. The hash value will serve establish mutual authentication between the server (gateway in our case) and the client (IoT edge device in our case).

Fig. 4 illustrates the key extraction process using the fuzzy extractor, also known as secure sketch.

At the server side, the key extractor block uses the low-entropy response w to generate a stable high-entropy key using any technique such as hashing for example as shown in Fig. 3.

The client receives the challenge c and helper data r and in response produces the actual noisy response w' and with the help of r it decodes w' to produce a copy of the error-free response w. The client then hashes this value and sends h' to the server to establish mutual authentication. Having regenerated the error-free response w, the server generates the secret key using the Key Generation block as shown in Fig. 3.

It should be noted that the secret key changes each time a new challenge c is issued. In this paper we will use this feature to generate a nonce which could be N_d or a hashed value of N_d to increase its entropy. This will serve to construct a secret key shared among the four entities of our system: mobile device (M), server (S), gateway (G) and IoT edge device (D).

Key extraction using the fuzzy extractor process can be expressed by the equation

\[ (K_{dg}, N_d) = \text{key\_regen}(c, r) \]  

(1)

where \( K_{dg} \) is the secret key and \( N_d \) is the secret random number.

Some implementations were done in FPGA platforms [36,51] and some were implemented on microcontrollers [52]. Gao et al. [49] proposed an SRAM-based PUF key generator on a microcontroller using radio frequency (RF) energy harvesting.

3.8. Gateway model

The gateway is located at the IoT home location or it could be located at a medical clinic at a geographically remote area. The compute resources at the gateway must be able to implement layered security starting from the applications all the way down to the hardware processors. A hardware root-of-trust (HRoT) could be implemented in order to secure the cryptographic keys and cryptographic primitives and protocols. The concomitant cost is justified by resulting enhanced security features.

A random number (nonce) \( N_d \) is generated at the gateway and applied to a cryptographic hash function to generate the secret key \( K_{dg} \) with a predetermined number of bits and high entropy. This serves two purposes, \( N_d \) serves to query the presence and freshness of the connection with the IoT device and \( K_{dg} \) serves to generate a one-time password (OTP) for use in cryptographic operations for the current session. The key generation using the fuzzy extractor process can be expressed by the equation

\[ (K_{dg}, r) = \text{key\_gen}(ID_d, c, N_d) \]  

(2)

where \( K_{dg} \) is the secret key and \( r \) is the helper data. The authentication process starts by the gateway to generate the quantities \( N_d, K_{dg} \) and \( r \).

The authenticating device also queries the dataset associated with the device using its identity ID_d and a selected challenge c to obtain a response w. \( N_d \) is then encoded using a linear block code such as BCH and the resulting redundant bit are XORed with the PUF response to generate the helper data r. The helper data is public and is transmitted, along with c, to the IoT edge device at the start of a session.

At the start of a session, the gateway computes a nonce \( N_d \) and chooses a challenge c to generate a one-time password/secret key \( K_{dg} \) according to Eq. (2). The gateway sends the chosen challenge and helper data to the edge device

\[ G \rightarrow E : M = (c, r, N_d) \]

(3)

The edge device is capable of generating its own copy of \( K_{dg} \) through the publicly received quantities c and r. At this stage, both the gateway and edge device know \( K_{dg} \) and \( N_d \) based on Eq. (1).
4. The proposed scheme

The gateway manages all communications between the edge devices and the server. Similarly, the server manages all communications with the gateway as well as with the mobile devices. The proposed protocol is divided into several phases.

1. Predeployment
2. Registration
3. Login
4. Authentication
5. Password update

4.1. Predeployment phase

4.1.1. Server

As was mentioned in Section 3.5, the server establishes secure communication with the RA through a symmetric secret key \( K_{s} \). The server is also assigned a unique identity \( ID_{s} \). The server can be considered a HRoT and implements several layers of security protocols. The server communicates with the RA to obtain credentials for the gateways, mobile devices and edge devices comprising the main components of the telehealth system in question.

4.1.2. Gateway

The manufacturer assigns to the gateway a symmetric secret key \( K_{gs} \) for communication with the server and also assigns a unique identity \( ID_{g} \).

4.1.3. Mobile device

The mobile device is considered here as the access device used by the healthcare professional to communicate with the IoT edge devices through the server and gateway. The manufacturer assigns to each mobile device a symmetric secret key \( K_{m} \) for communication with the server and also assigns a unique identity \( ID_{m} \). A password \( PW_{m} \) and a biometric \( B_{m} \) will be supplied by the user.

4.1.4. Edge device

The manufacturer assigns a unique ID during manufacture (\( ID_{d} \)) and after the device is fabricated, the manufacturer generates the device CRP Dataset which must be kept secret to be shared later by an authenticating entity. The fabricator also includes in the CRP dataset the helper data as per Section 3.7.

4.2. Registration phase

The main parties involved in the operation of the telehealth system are the registration authority (RA), the server and the device fabricator (fab house) responsible for manufacturing the gateway, mobile devices, and edge devices. The server and the fab house establish secure communication with the RA through public key infrastructure (PKI).

The administrator for the telehealth system establishes associations between

1. The server and the gateway
2. The server and the mobile devices connected to it
3. The gateway and the edge devices connected to it

Fig. 5 shows the registration phase implemented by the manufacturer (fab house) for the gateway, mobile devices and edge devices, as will be explained below.

The following transactions take place.

Transaction #1: the server contacts the RA requesting data associated with the gateway, mobile devices and edge devices.
Transaction #2: the RA sends the server all the data associated with the gateway, mobile devices and edge devices.

Transaction #3: the server sends the gateway all the data associated with the edge devices connected to it. Note that in our protocol the gateway only communicates with the edge devices and server. The gateway, mobile devices, or edge devices are not allowed to directly contact the RA or the mobile devices directly to reduce the chance of attacks.

4.2.1. Registration of gateway

Registration of the gateway is initiated by the server through communicating with the RA to obtain the gateway secret key $K_{G}$. Communication starts with a challenge message from $S$ and a response from RA.

$$S : m_{g1} = E(K_{sr}, (ID_{s}||ID_{G}||N_{s1}))$$ (4)

$$S \rightarrow RA : Request(ID_{s}, ID_{ra}, m_{g1})$$ (5)

$$RA : V_{g} = h(ID_{s}||ID_{G}||K_{G}||N_{s1})$$ (6)

$$RA \rightarrow S : E(K_{sr}, (K_{G}||ID_{G}||V_{g}||N_{s1}))$$ (7)

$$S : m_{g2} = E(K_{sr}, (ID_{G}||V_{g}||N_{s2}))$$ (8)

$$S \rightarrow G : Request(ID_{s}, ID_{G}, m_{g2})$$ (9)

$$G \rightarrow S : E(K_{G}, N_{s2})$$ (10)

where $N_{s1}$ and $N_{s2}$ are nonces to ensure message freshness and the hash $V_{g}$ will be used later in the authentication phase between $S$ and $G$.

Operation in Eq. (4): server prepares a challenge for RA as an encrypted message that includes a nonce $N_{s1}$.

Operation in Eq. (5): server sends a request to communicate with RA.

Operation in Eq. (6): RA prepares the hash $V_{g}$ to be used later for authentication between $S$ and $G$.

Operation in Eq. (7): RA responds by sending back the nonce as a proof of existence and freshness.

Operation in Eq. (8): server prepares a challenge for $G$ as an encrypted message that includes the nonce $N_{s2}$.

Operation in Eq. (9): server sends a request to communicate with $G$.

Operation in Eq. (10): gateway responds to the challenge by sending back the encrypted nonce $N_{s2}$.

4.2.2. Registration of the mobile device

Registration of the mobile device $M$ is initiated by the mobile device through communicating with $S$ that in turn communicates with the RA to obtain the mobile device identity $ID_{m}$. The server also informs RA of the password selected by the user $PW_{m}$.

$$M : m_{m1} = E(K_{ms}, (ID_{m}||N_{m}))$$ (11)

$$M \rightarrow S : Request(ID_{m}, ID_{s}, m_{m1})$$ (12)

$$S : m_{m2} = E(K_{sr}, (ID_{s}||ID_{m}||N_{s1}))$$ (13)

$$S \rightarrow RA : Request(ID_{s}, ID_{ra}, m_{2})$$ (14)

$$RA : V_{m} = h(ID_{m}||PW_{m})$$ (15)

$$RA \rightarrow S : E(K_{sr}, (ID_{m}||V_{m}||N_{s1}))$$ (16)

$$S \rightarrow M : E(K_{ms}, (ID_{m}||V_{m}||N_{s2})||m_{m1})$$ (17)

$$M \rightarrow S : E(K_{ms}, N_{s2})$$ (18)

where the hash $V_{m}$ will be used later in the authentication phase between $S$ and $M$. The steps in Eq. (11)–Eq. (18) have the same meaning as was explained for steps Eq. (4)–Eq. (10).
4.2.3. Registration of the edge device

The network administrator informs the server of all the entities forming the telehealth system including server, mobile devices, gateway, and all the edge devices connected to the gateway. As a result, registration of the edge devices is initiated by the server through communicating with the RA to obtain each edge device secret key $K_{dg}$ and CRP$_d$ Dataset, corresponding to the built-in PUF, as well as fuzzy extractor data such as BCH code with parameters $(n, k, d)$ as well the generator matrix $(G)$ and parity check matrix $(H)$. The server also communicates these information to the gateway.

$$S : m_1 = E(K_{sr}, \ (ID_s||ID_g||ID_d||N_{s1}))$$ (19)

$$S \rightarrow RA : \text{Request}(ID_s, \ ID_{ra}, \ m_1)$$ (20)

$$RA : V_d = h(ID_d||K_{dg}||N_{s1})$$ (21)

$$RA : m_2 = TID_d || G || H || CRP_d || (n, k, d) || V_d || N_{s1}$$ (22)

$$RA \rightarrow S : E(K_{sr}, \ m_2)$$ (23)

$$S : m_3 = E(K_{gs}, \ (ID_d||V_d||N_{s2}))$$ (24)

$$S \rightarrow G : \text{Request}(ID_s, \ ID_{gs}, \ m_3)$$ (25)

$$G \rightarrow S : E(K_{gs}, \ N_{s2})$$ (26)

$$G : m_4 = E(K_{dg}, \ (ID_d||V_d||N_{g}))$$ (27)

$$G \rightarrow E : \text{Request}(ID_d, \ ID_{dg}, \ m_4)$$ (28)

$$D \rightarrow G : E(K_{dg}, \ N_{g})$$ (29)

where the hash $V_d$ and $TID_d$ will to be used for the duration of the session during communication between the gateway and the edge device. The steps in Eqs. (19)–(29) have the same meaning as was explained for steps Eqs. (4)–(10).

4.3. Login phase

The healthcare professional logs in to the telehealth system through the mobile device using three-factor authentication by what he/she has (mobile device), what he/she knows (password), and what he/she is (biometric). The mobile device also computes $V'_m$ and sends it to the server.

$$M : V'_m = h(ID_m||PW_m)$$ (30)

$$S : V_m = = = V'_m$$ (31)

Login is successful when $V_m$ matches $V'_m$. If first login is not successful, another login is established by asking for other personal information such as a security question or reaching out to the system administrator. A limited number of login attempts is allowed and if this number is exceeded, the device terminates the login request immediately until the user re-registers again.
4.4. Authentication phase

Mutual authentication is required between the communicating entities. We consider here the case when a healthcare professional desires to communicate with an edge device. In that case four entities are involved: mobile device M, server S, gateway G and finally the edge device D. Fig. 6 shows the authentication phase between the four communicating entities. Authentication proceeds as three stages:

1. Mobile device–server stage
2. Server–gateway stage
3. Gateway–edge device stage

4.4.1. Mobile device–server stage

The mobile device chooses a nonce $N_1$ and obtains its current location $L_m$ and calculates a dynamic identity

$$DID_m = ID_m \oplus N_1$$

(32)

The dynamic identity depends on the nonce $N_1$ and becomes unique to each session to preserve anonymity and untraceability.

The mobile device employs hash chains to calculate the quantities:

$$MS = (DID_m||N_1||L_m) \oplus V_m$$

(33)

$$H_{ms} = h(DID_m||N_1||L_m)$$

(34)

where $V_m$ was obtained from RA in Eq. (15).

The mobile device sends the following message to the server

$$M \rightarrow S : (MS, H_{ms})$$

(35)

This assures the server of the freshness and presence of the mobile device.

The server receives the message in Eq. (35), and computes $MS$ using the stored value $V_m$

$$DID_m||N_1||L_m = MS \oplus V_m$$

(36)

The server checks the freshness of received nonce $N_1$ to prevent replay attack and computes $ID^*_m$ and $H^*_{ms}$

$$ID^*_m = DID_m \oplus N_1$$

(37)

$$H^*_{ms} = h(DID_m||N_1||L_m)$$

(38)

The mobile device is authenticated when the following equalities are satisfied:

$$ID^*_m = ID_m$$

$$H^*_{ms} = H_{ms}$$

$$L^*_m \leq L_{m-1} + \Delta$$

where $\Delta$ is the maximum change allowed in the location between two sessions. This verifies the integrity of the message, otherwise the server will terminate the session with the mobile device.

4.4.2. Server–gateway stage

The server prepares a message to send to the gateway. It starts by generating a nonce $N_2$ and computes a server dynamic identity $DID_s$:

$$DID_s = ID_s \oplus N_2$$

(39)

The server then computes:

$$SG = (DID_s||N_1||N_2) \oplus V_g$$

(40)

$$H_{sg} = h(DID_s||N_1||N_2)$$

(41)
The server sends the following message to the gateway

$$S \rightarrow G : (SG, H_{sg})$$  \hspace{1cm} (42)

The gateway receives the message in Eq. (42) from the server, and computes $SG$ using the stored value $V_g$

$$DID_s || N_1 || N_2 = SG \oplus V_g$$  \hspace{1cm} (43)

The gateway computes $ID^*_s$ and $H^*_{sg}$

$$ID^*_s = DID_s \oplus N_2$$  \hspace{1cm} (44)

$$H^*_{sg} = h(DID_s || N_1 || N_2)$$  \hspace{1cm} (45)

The gateway compares $H^*_{sg}$ with the received value $H_{sg}$. This verifies the integrity of the message, otherwise the gateway will terminate the session with the server.

4.4.3. Gateway–edge device stage

The gateway prepares a message to send to the edge device. It starts by preparing a nonce $N_3$ and computes

$$GE = (N_1 || N_2 || N_3) \oplus V_d$$  \hspace{1cm} (46)

$$H_{ge} = h(N_1 || N_2 || N_3)$$  \hspace{1cm} (47)

The gateway sends the following message to the edge device

$$G \rightarrow E : (GE, H_{ge})$$  \hspace{1cm} (48)

The edge device receives the message in Eq. (48) from the gateway, and computes $GD$ using the stored value $V_d$

$$N_1 || N_2 || N_3 = GD \oplus V_d$$  \hspace{1cm} (49)

The edge device computes $H^*_{ge}$

$$H^*_{ge} = h(N_1 || N_2 || N_3)$$  \hspace{1cm} (50)

The edge device compares $H^*_{ge}$ with the received value $H_{ge}$. This verifies the integrity of the message, otherwise the edge device will terminate the session with the gateway.

4.4.4. The reverse path

The edge device generates a nonce $N_4$ and computes its dynamic identity $DID_d$ and the session key $SK$

$$DID_d = ID_d \oplus N_4$$  \hspace{1cm} (51)

$$SK = h(N_1 || N_2 || N_3 || N_4)$$  \hspace{1cm} (52)

The edge device collects the identities of all the edge devices it sees around it through its Bluetooth or Zigbee connection:

$$\mathbb{D}_d = \{ID_d | i \in \mathbb{D}\}$$  \hspace{1cm} (53)

where $\mathbb{D}_d$ is the set of identities of all edge devices seen by the particular edge device being authenticated and $\mathbb{E}$ is the set of all edge devices in the IoT network.

The edge device prepares a reply to the gateway by computing the following quantities

$$DG = (DID_d || \mathbb{D}_d || N_4) \oplus V_d$$  \hspace{1cm} (54)

$$H_{gd} = h(N_4 || SK)$$  \hspace{1cm} (55)

The edge device sends the following message to the gateway

$$D \rightarrow G : (DG, H_{gd})$$  \hspace{1cm} (56)

When the gateway receives the message from the edge device, it will extract $N_4$ and $\mathbb{D}_d$:

$$(DID_d || \mathbb{D}_d || N_4) = DG \oplus V_g$$  \hspace{1cm} (57)

The gateway then computes the device identity

$$ID_d = DID_d \oplus N_4$$  \hspace{1cm} (58)
The edge device is authenticated when the following are satisfied
\[
ID_g^* == ID_d \\
D_d \neq \phi \\
D_d \subset D \\
H_{gd}^* == H_{gd}
\]

Having the value \( N_4 \), the gateway verifies message integrity using Eq. (55) and ensures the validity of the above equations. The gateway now has the value \( N_4 \) and can independently calculate its dynamic identity and the session key \( SK \) using Eq. (52).

\[
DID_g^* = ID_g \oplus N_3 \tag{59}
\]

The gateway embeds the values \( N_3 \) and \( N_4 \) in the following quantity
\[
GS = (DID_g^*||N_3||N_4) \oplus V_g \tag{60}
\]

The gateway also computes \( H_{gs} \)

\[
H_{gs} = h(N_3||N_4||SK) \tag{61}
\]

The gateway now sends the following message to the server

\[
G \rightarrow S : (GS||H_{gs}) \tag{62}
\]

When the server receives the message from the gateway, it will extract \( N_3 \) and \( N_4 \)

\[
(DID_g^*||N_3||N_4) = GS \oplus V_g \tag{63}
\]

Using \( N_3 \) and \( N_4 \), the server will independently calculate the gateway dynamic identity \( DID_g^* \) using Eq. (59) and the session key \( SK \) using Eq. (52).

The server verifies the integrity of the message by calculating \( H_{gs}^* \) form Eq. (61) and compare it with the received \( H_{gs} \).

The server now has the values \( N_2, N_3 \) and \( N_4 \). The server now embeds the values \( N_2, N_3 \) and \( N_4 \) in the following quantity

\[
SM = (N_2||N_3||N_4) \oplus V_m \tag{64}
\]

The server also computes \( H_{sm} \)

\[
H_{sm} = h(N_2||N_3||N_4) \tag{65}
\]

The server now sends the following message to the mobile device

\[
S \rightarrow M : (SM||H_{sm}) \tag{66}
\]

When the mobile device receives the message from the server, it will extract \( N_2, N_3 \) and \( N_4 \)

\[
(N_2||N_3||N_4) = SM \oplus V_m \tag{67}
\]

Using \( N_2, N_3 \) and \( N_4 \), the mobile device will independently calculate the session key \( SK \) using Eq. (52).

The mobile device verifies the integrity of the message by calculating \( H_{sm}^* \) form Eq. (65) and compare it with the received \( H_{sm} \).

### 4.5. Password update phase

The password update phase applies to the user of the mobile device. The user can update the password after registration without involving the RA. In order to do that, the user supplies \( ID_m \) and the old password. The mobile device calculates \( V_m^* \) according to Eq. (15) and compare this with \( V_m \) that was defined during the registration phase. If the two values match, then the password update process can proceed further. The mobile device calculates a new \( V_m \) based on the new password and send this to the server.

### 5. Security analysis of the proposed scheme

To analyze the security of our proposed scheme, we use BAN logic [53] and formal model checking and simulations using AVISPA tool [54] and informal security analysis.
### Table 3

Notations in BAN logic.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$ and $Q$</td>
<td>Principals</td>
</tr>
<tr>
<td>$P</td>
<td>\equiv X$</td>
</tr>
<tr>
<td>$P \vdash X$</td>
<td>Principal $P$ sees the statement $X$</td>
</tr>
<tr>
<td>$P</td>
<td>\Rightarrow X$</td>
</tr>
<tr>
<td>$P</td>
<td>\sim X$</td>
</tr>
<tr>
<td>$(X, Y)$</td>
<td>The statement $X$ or $Y$ is one part of message $(X, Y)$</td>
</tr>
<tr>
<td>$&lt; X \sim_f$</td>
<td>The statement $X$ is encrypted with the key $K$</td>
</tr>
<tr>
<td>$(X)_K$</td>
<td>The statement $X$ is hashed with the key $K$</td>
</tr>
<tr>
<td>$P \leftrightarrow Q$</td>
<td>$K$ is a secret parameter shared (or to be shared) between $P$ and $Q$</td>
</tr>
<tr>
<td>$P \equiv Q$</td>
<td>$X$ is a secret known only to $P$ and $Q$, and possibly to parties trusted by them.</td>
</tr>
<tr>
<td>$#(X)$</td>
<td>The message $X$ is fresh.</td>
</tr>
</tbody>
</table>

5.1. Formal proof based on BAN logic

In this subsection, we introduce a formal analysis for the proposed scheme using widely accepted model called BAN logic, this model has been used for a formal verification of security protocols which introduced in 1989 by Burrows et al. [53]. We begin our analysis by introducing the most important symbols and notations adapted from [20] which are given in Table 3.

In addition, the following BAN logic basic rules are used to prove that our authentication protocol provides secure mutual authentication and key agreement as follows:

- **Message-meaning rule:**
  If $P$ believes that the key $K$ is shared with $Q$ and $P$ sees $X$ encrypted under $K$, then $P$ believes that $Q$ once said $X$.

  $P| \equiv Q \leftrightarrow K, P \vdash (X)_K$

  $\Rightarrow P| \equiv Q| \sim X$

- **Nonce verification rule:**
  If $P$ believes $X$ is fresh and $P$ believes $Q$ once said $X$, then $P$ believes $Q$ believes $X$.

  $P| \equiv #(X), P| \equiv Q| \sim X$

  $\Rightarrow P| \equiv Q| \equiv X$

- **Jurisdiction rule:**
  If $P$ believes $Q$ has jurisdiction over $X$ and $P$ believes $Q$ believes $X$, then $P$ believes $X$.

  $P| \equiv Q| \Rightarrow X, P| \equiv Q| \equiv X$

  $\Rightarrow P| \equiv X$

- **Freshness conjunction rule:**
  If one part of a statement is fresh, then the entire statement must also be fresh; so if $P$ believes $X$ is fresh, then $P$ believes $X$ and $Y$ are fresh.

  $P| \equiv #(X)$

  $\Rightarrow P| \equiv #(X,Y)$

- **Belief rule:** If $P$ believes $X$ and $Y$, then $P$ believes $X$.

  $P| \equiv (X, Y)$

  $\Rightarrow P| \equiv X$

- **Session keys rule:**

  $P| \equiv #(X), P| \equiv Q| \equiv X$

  $\Rightarrow P| \equiv P \leftrightarrow K Q$

The proposed scheme must achieve the following goals:

- **Goal 1**

  $S| \equiv M| \equiv M^{SK} S$

- **Goal 2:**

  $S| \equiv M^{SK} S$
• Goal 3:
  \[ G | \equiv S | \equiv S^{SK} G \]
• Goal 4:
  \[ G | \equiv S^{SK} G \]
• Goal 5:
  \[ D | \equiv G | \equiv G^{SK} D \]
• Goal 6:
  \[ D | \equiv G^{SK} D \]
• Goal 7:
  \[ G | \equiv D | \equiv G^{SK} D \]
• Goal 8:
  \[ G | \equiv G^{SK} D \]
• Goal 9:
  \[ S | \equiv G | \equiv S^{SK} G \]
• Goal 10:
  \[ S | \equiv S^{SK} G \]
• Goal 11:
  \[ M | \equiv S | \equiv M^{SK} S \]
• Goal 12:
  \[ M | \equiv M^{SK} S \]
• Goal 13:
  \[ M | \equiv G | \equiv M^{SK} G \]
• Goal 14:
  \[ G | \equiv M | \equiv M^{SK} G \]
• Goal 15:
  \[ S | \equiv D | \equiv S^{SK} D \]
• Goal 16:
  \[ D | \equiv S | \equiv S^{SK} D \]

The fundamental assumptions of the authentication protocol are as follows:

• A1:
  \[ M | \equiv \#(N_2) \]
• A2:
  \[ M | \equiv \#(N_3) \]
• A3:
  \[ M | \equiv \#(N_4) \]
• A4:
  \[ S | \equiv \#(N_1) \]
• A5:
  \[ S | \equiv \#(N_3) \]
• A6: 
  \( S \equiv \#(N_4) \)
• A7: 
  \( G \equiv \#(N_2) \)
• A8: 
  \( G \equiv \#(N_4) \)
• A9: 
  \( D \equiv \#(N_1) \)
• A10: 
  \( D \equiv \#(N_2) \)
• A11: 
  \( D \equiv \#(N_3) \)
• A12: 
  \( S \equiv M \overset{V_m}{\leftrightarrow} S \)
• A13: 
  \( G \equiv S \overset{V_g}{\leftrightarrow} G \)
• A14: 
  \( D \equiv G \overset{V_g}{\leftrightarrow} D \)
• A15: 
  \( G \equiv D \overset{V_g}{\leftrightarrow} G \)
• A16: 
  \( M \equiv S \overset{V_m}{\leftrightarrow} M \)
• A17: 
  \( S \equiv G \overset{V_g}{\leftrightarrow} S \)
• A18: 
  \( G \equiv D \Rightarrow (N_4, ID_d, V_d, SK) \)
• A19: 
  \( S \equiv G \Rightarrow (N_3, N_4, ID_g, V_g, SK) \)
• A20: 
  \( M \equiv S \Rightarrow (N_2, N_3, N_4, V_m, SK) \)
• A21: 
  \( D \equiv G \Rightarrow (N_1, N_2, N_3, V_d, SK) \)
• A22: 
  \( S \equiv M \Rightarrow (N_1, ID_m, V_m, L_m, SK) \)
• A23: 
  \( G \equiv S \Rightarrow (N_1, N_2, ID_g, V_g, SK) \)

Messages transferred in the authentication protocol:

• **Msg 1:** 
  \( M \rightarrow S : (MS, H_{ms}) \overset{V_m}{M \rightarrow S} \)
• **Msg 2:** 
  \( S \rightarrow G : (SG, H_{sg}) \overset{V_g}{S \rightarrow G} \)
- **Msg 3:**
  \[ G \rightarrow D : (GD, H_{gd}) \]  
  \[ G \rightarrow D : (GD, H_{gd})_{G\rightarrow D} \]  

- **Msg 4:**
  \[ D \rightarrow G : (DG, H_{dg}) \]  
  \[ D \rightarrow G : (DG, H_{dg})_{D\rightarrow G} \]  

- **Msg 5:**
  \[ G \rightarrow S : (GS, H_{gs}) \]  
  \[ G \rightarrow S : (GS, H_{gs})_{G\rightarrow S} \]  

- **Msg 6:**
  \[ S \rightarrow M : (SM, H_{sm}) \]  
  \[ S \rightarrow M : (SM, H_{sm})_{S\rightarrow M} \]  

Analysis of our authentication scheme:

- **S1:** According to Msg 1, we get:
  \[ S \leftarrow (MS, H_{ms})_{M\leftarrow S} \]  

- **S2:** Based on Assumption A12, S1 and message-meaning rule, we have:
  \[ S | \equiv M \vdash S, S \leftarrow (MS, H_{ms})_{M\leftarrow S} \]  
  \[ S | \equiv M | \sim (MS, H_{ms})_{M\leftarrow S} \]  

- **S3:** From A4 and freshness-conjunctcatenation rule, we get:
  \[ S | \equiv \# (MS, H_{ms})_{M\leftarrow S} \]  

- **S4:** From S3, S2 and nonce-verification rule, we get:
  \[ S | \equiv \# (MS, H_{ms})_{M\leftarrow S} \sim M | \equiv (MS, H_{ms}) \]  
  \[ S | \equiv M | \equiv (MS, H_{ms})_{M\leftarrow S} \]  

- **S5:** According to the Msg 2, we get:
  \[ G \leftarrow (SG, H_{sg})_{S\leftarrow G} \]  

- **S6:** From A13, S5 and message-meaning rule, we have:
  \[ G | \equiv S \vdash G, G \leftarrow (SG, H_{sg})_{S\leftarrow G} \]  
  \[ G | \equiv S | \sim (SG, H_{sg})_{S\leftarrow G} \]  

- **S7:** From A7 and freshness-conjunctcatenation rule, we get:
  \[ G | \equiv \# (SG, H_{sg})_{S\leftarrow G} \]  

- **S8:** From S7, S6 and nonce-verification rule, we get:
  \[ G | \equiv \# (SG, H_{sg})_{S\leftarrow G} \sim G | \equiv (SG, H_{sg})_{S\leftarrow G} \]  
  \[ G | \equiv S | \equiv (SG, H_{sg})_{S\leftarrow G} \]  

- **S9:** According to the Msg 3, we get:
  \[ D \leftarrow (GD, H_{gd})_{G\leftarrow D} \]  

- **S10:** From A14, S9 and message-meaning rule, we have:
  \[ D | \equiv (G \vdash D), D \leftarrow (GD, H_{gd})_{G\leftarrow D} \]  
  \[ D | \equiv G | \sim (GD, H_{gd})_{G\leftarrow D} \]  

- **S11:** From A11 and freshness-conjunctcatenation rule, we get:
  \[ D | \equiv \# (GD, H_{gd})_{G\leftarrow D} \]  

- **S12:** From S11, S10 and nonce-verification rule, we get:
  \[ D | \equiv \# (GD, H_{gd})_{G\leftarrow D} \sim D | \equiv G | \sim (GD, H_{gd})_{G\leftarrow D} \]  
  \[ D | \equiv G | \equiv (GD, H_{gd})_{G\leftarrow D} \]  

- **S13:** According to the Msg 4, we get:
  \[ G \leftarrow (DG, H_{dg})_{G\leftarrow D} \]
• S14: From A15, S13 and message-meaning rule, we have:
  \[ G \equiv D \leftrightarrow G, G \circ (DG, H_{dg})_{G_S D} \]
  \[ G \equiv D \sim (DG, H_{dg})_{G_S D} \]
  \[ G \equiv D \equiv (DG, H_{dg})_{G_S D} \]

• S15: From A8 and freshness-conjunctagation rule, we get:
  \[ G \equiv # (DG, H_{dg})_{G_S D} \]

• S16: From S15, S14 and nonce-verification rule, we get:
  \[ G \equiv # (DG, H_{dg})_{G_S D}, G \equiv D \sim (DG, H_{dg})_{G_S D} \]
  \[ G \equiv D \equiv (DG, H_{dg})_{G_S D} \]

• S17: According to the Msg 5, we get:
  \[ S \circ (GS, H_{GS})_{S^G G} \]

• S18: From A17, S17 and message-meaning rule, we have:
  \[ S| \equiv G \leftrightarrow S, S \circ (GS, H_{GS})_{S^G G} \]
  \[ S| \equiv G| \sim (GS, H_{GS})_{S^G G} \]

• S19: From A5, A6 and freshness-conjunctagation rule, we get:
  \[ S| \equiv # (GS, H_{GS})_{S^G G} \]

• S20: From S19, S18 and nonce-verification rule, we get:
  \[ S| \equiv # (GS, H_{GS})_{S^G G}, S| \equiv G| \sim (GS, H_{GS})_{S^G G} \]
  \[ S| \equiv G| \equiv (GS, H_{GS})_{S^G G} \]

• S21: According to the Msg 6, we get:
  \[ M \circ (SM, H_{SM})_{S^M M} \]

• S22: From A16, S21 and message-meaning rule, we have:
  \[ M| \equiv S \leftrightarrow M, M \circ (SM, H_{SM})_{S^M M} \]
  \[ M| \equiv S| \sim (SM, H_{SM})_{S^M M} \]

• S23: From A1, A2, A3 and freshness-conjunctagation rule, we get:
  \[ M| \equiv # (SM, H_{SM})_{S^M M} \]

• S24: From S23, S22 and nonce-verification rule, we get:
  \[ M| \equiv # (SM, H_{SM})_{S^M M}, M| \equiv S| \sim (SM, H_{SM})_{S^M M} \]
  \[ M| \equiv S| \equiv (SM, H_{SM})_{S^M M} \]

• S25: From A22, S4 and jurisdiction rule, we get:
  \[ S| \equiv M| \Rightarrow (N_1, ID_m, V_m, L_m, SK), S| \equiv M| \equiv (MS, H_{ms})_{M^S S} \]
  \[ S| \equiv (MS, H_{ms})_{M^S S} \]

• S26: From S3, S4 and session keys rule, we get:
  \[ S| \equiv # (MS, H_{ms})_{M^S S}, S| \equiv M| \equiv (MS, H_{ms})_{M^S S} \]
  \[ S| \equiv M| \equiv M \leftrightarrow S \]
  \[ (Goal 1) \]

• S27: From A22, S26 and jurisdiction rule, we get:
  \[ S| \equiv M| \Rightarrow (N_1, ID_m, V_m, L_m, SK), S| \equiv M| \equiv M \leftrightarrow S \]
  \[ S| \equiv M \leftrightarrow S \]
  \[ (Goal 2) \]
• S28: From A23, S8 and jurisdiction rule, we get:
\[
G | \equiv S \Rightarrow (N_1, N_2, ID, V_g, SK), G | \equiv S | \equiv (SG, H_{sk})_{S^G}^{\nu_G}
\]
\[
G | \equiv (SG, H_{sk})_{S^G}^{\nu_G}
\]

• S29: From S7, S8 and session keys rule, we get:
\[
G | \equiv # (SG, H_{sk})_{S^G}^{\nu_G}, G | \equiv S | \equiv (SG, H_{sk})_{S^G}^{\nu_G}
\]
\[
G | \equiv S | \equiv S^{SK} \Leftrightarrow G
\]

(Goal 3)

• S30: From A23, S29 and jurisdiction rule, we get:
\[
G | \equiv S \Rightarrow (N_1, N_2, ID, V_g, SK), G | \equiv S | \equiv S^{SK} \Leftrightarrow G
\]
\[
G | \equiv S^{SK} \Leftrightarrow G
\]

(Goal 4)

• S31: From A21, S12 and jurisdiction rule, we get:
\[
D | \equiv G | \Rightarrow (N_1, N_2, N_3, V_d, SK), D | \equiv G | \equiv (GD, H_{gd})_{G^D}^{\nu_D}
\]
\[
D | \equiv (GD, H_{gd})_{G^D}^{\nu_D}
\]

• S32: From S11, S12 and session keys rule, we get:
\[
D | \equiv # (GD, H_{gd})_{G^D}^{\nu_D}, D | \equiv G | \equiv (GD, H_{gd})_{G^D}^{\nu_D}
\]
\[
D | \equiv G | \equiv G^{SK} \Leftrightarrow D
\]

(Goal 5)

• S33: From A21, S32 and jurisdiction rule, we get:
\[
D | \equiv G | \Rightarrow (N_1, N_2, N_3, V_d, SK), D | \equiv G | \equiv G^{SK} \Leftrightarrow D
\]
\[
D | \equiv G^{SK} \Leftrightarrow D
\]

(Goal 6)

• S34: From A18, S16 and jurisdiction rule, we get:
\[
G | \equiv D | \Rightarrow (N_4, ID_d, V_d, SK), G | \equiv D | \equiv (DG, H_{dg})_{G^D}^{\nu_D}
\]
\[
G | \equiv (DG, H_{dg})_{G^D}^{\nu_D}
\]

• S35: From S15, S16 and session keys rule, we get:
\[
G | \equiv # (DG, H_{dg})_{G^D}^{\nu_D}, G | \equiv D | \equiv (DG, H_{dg})_{G^D}^{\nu_D}
\]
\[
G | \equiv D | \equiv G^{SK} \Leftrightarrow D
\]

(Goal 7)

• S36: From A18, S35 and jurisdiction rule, we get:
\[
G | \equiv D | \Rightarrow (N_4, ID_d, V_d, SK), G | \equiv D | \equiv G^{SK} \Leftrightarrow D
\]
\[
G | \equiv G^{SK} \Leftrightarrow D
\]

(Goal 8)

• S37: From A19, S20 and jurisdiction rule, we get:
\[
S | \equiv G | \Rightarrow (N_3, N_4, ID_g, V_g, SK), S | \equiv G | \equiv (GS, H_{gs})_{G^G}^{\nu_G}
\]
\[
S | \equiv (GS, H_{gs})_{G^G}^{\nu_G}
\]

• S38: From S19 and S20, we get:
\[
S | \equiv # (GS, H_{gs})_{G^G}^{\nu_G}, S | \equiv G | \equiv (GS, H_{gs})_{G^G}^{\nu_G}
\]
\[
S | \equiv G | \equiv S^{SK} \Leftrightarrow G
\]

(Goal 9)

• S39: From A19, S38 and jurisdiction rule, we get:
\[
S | \equiv G | \Rightarrow (N_3, N_4, ID_g, V_g, SK), S | \equiv G | \equiv S^{SK} \Leftrightarrow G
\]
\[
S | \equiv S^{SK} \Leftrightarrow G
\]

(Goal 10)
• S40: From A20, S24 and jurisdiction rule, we get:

\[
M| S \Rightarrow (N_2, N_3, N_4, V_m, SK), M| S \equiv (SM, H_{SM})_{M_{20S}}
\]

\[
M \equiv (SM, H_{SM})_{M_{20S}}
\]

• S41: From S23, S24 and session keys rule, we get:

\[
M| S \Rightarrow (SM, H_{SM})_{S_{20M}}, M| S \equiv (SM, H_{SM})_{S_{20M}}
\]

\[
M| S \equiv M^{SK} S
\]

(Goal 11)

• S42: From A20, S41 and jurisdiction rule, we get:

\[
M| S \Rightarrow (N_2, N_3, N_4, V_m, SK), M| S \equiv M^{SK} S
\]

\[
M| S \equiv M^{SK} S
\]

(Goal 12)

• S43: From S41, S30, we get:

\[
M| S \equiv M^{SK} S, G| S \equiv S^{SK} G
\]

\[
M| G \equiv M^{SK} G
\]

(Goal 13)

• S44: From S29, S42 and jurisdiction rule, we get:

\[
G| S \equiv S^{SK} G, M| S \equiv M^{SK} S
\]

\[
G\equiv M| S \equiv M^{SK} G
\]

(Goal 14)

• S45: From S38, S33 and jurisdiction rule, we get:

\[
S| G \equiv S^{SK} G, D| G \equiv G^{SK} D
\]

\[
S| D \equiv S^{SK} D
\]

(Goal 15)

• S46: From S32, S39 and jurisdiction rule, we get:

\[
D| G \equiv G^{SK} D, S| G \equiv S^{SK} G
\]

\[
D| S \equiv S^{SK} D
\]

(Goal 16)

Hence, the above BAN logic analysis formally proves that the proposed scheme successfully achieves mutual authentication, and the session key SK, in Eq. (52), is mutually established between the U and the D_j through the G.

5.2. Informal security analysis

In this section, we show how our protocol is robust against various well-known attacks.

5.2.1. Replay attack

To resist replay attack, the random number method is utilized, so replay attack can be prevented by using the nonces which change in each session.

5.2.2. Eavesdropping attack

In our scheme, the adversary can easily intercept the messages between M, S, G and D since all messages are sent in plain text. However, the attacker cannot access sensitive information from any messages because the confidential data is secured by secret parameters shared securely between the entities (e.g., \(V_m\) and \(V_k\)), and shielded by one-way hash function and bitwise XOR operator. The attacker would thus not be able to unfold the transmitted parameters, and thus can not extract any useful information.

5.2.3. Impersonation attack

Assume an attacker A attempts to impersonate a healthcare professional. A can not succeed because he/she does not know the password or the biometric needed for three-factor authentication to access the mobile device.
5.2.4. Man-in-the-middle attack

The proposed scheme offers mutual authentication, as stated in Section 5.1 of the BAN logic. The transmitted messages are further protected by the secret values and nonces, and no one can forge legally authenticated messages without knowing those secret values. The proposed scheme hence prevents the Man-in-the-Middle attack.

5.2.5. Forward/backward secrecy

The session key is built using four different random numbers that are generated by M, S, G and D in each session. Thus, if the session key SK is compromised by an attacker, the confidential information of past or future communication sessions can not be compromised. For this reason, the proposed scheme achieves forward/backward secrecy.

5.2.6. Session key guessing attack

The session key SK is constructed by all communication participants, namely M, S, G and D, and four randomly selected nonces. Thus security relies on the randomness of the input values, which makes it difficult for an attacker to guess. An attacker’s probability of guessing the right key SK is negligible, provided that \(N_1, N_2, N_3\) and \(N_4\) are chosen randomly in every session.

5.2.7. User anonymity and untraceability

Anonymity of users and untraceability in authentication are two essential security properties. Anonymity ensures that the mobile device’s real identity is maintained secure, and that the mobile device stays unidentifiable among other devices. Therefore, the attacker is unable to identify the devices’ identities. Untraceability, on the other hand, means that the various sessions set up by a specific mobile device are not traceable, so that an attacker cannot relate any sessions to a specific mobile device. These two main security properties were achieved by the use of the healthcare professional’s dynamic identity, where we use different ID in each session.

5.2.8. Location-based authentication

The physical context awareness (location in the IoT system) used in our scheme involves checking the identities of the edge devices seen by the device D being authenticated, see Eq. (53). If the subset \(\mathcal{D}_d\) is valid and does not contain identities of unknown devices, then the location of our device is authenticated.

5.2.9. Cloning attack

Cloning attack targets the unprotected IoT edge devices. Section 3.1 discussed incorporating PUF modules in the edge devices which provides a high degree of tamper-resistant unique identity (fingerprint) without incurring extra costs in area, delay, power or specialized processing steps [41]. The unique device identity avoids using nonvolatile memory, whose contents can be easily obtained by an unsophisticated attacker. Instead the device ID is captured in the PUF circuit which provides a random response with low entropy that imparts sufficient differences between the manufactured edge devices. Therefore, IoT edge devices are immune to cloning attacks.

5.2.10. Physical attack

Physical attack attempts to obtain the secret key of the device knowing that secret keys are typically stored in nonvolatile memories. The IoT edge devices are the most vulnerable to this type of attack since they are usually located in unsecured locations. We mentioned above in Section 3.1 PUF modules are installed in the edge devices which provides a high degree of tamper-resistant unique identity (fingerprint). The PUF response is used to extract the secret key instead of relying on nonvolatile memories. Section 3.7 discussed how the secret key is extract from the noisy response of the PUF. Therefore, IoT edge devices are immune to physical attacks.

5.3. Simulation based on AVISPA tool

The proposed scheme was formally validated by AVISPA that is a widely used tool for security protocol assessments. The message exchanges and entities were defined in the high-level property specification language (HLPSL). The definitions and information in details can be found in [55].

This subsection explores several roles for system entities, the session, the goal and the environment of proposed scheme. In Figs. 7–10, we illustrate HLPSL code for our proposed scheme. These figures show the HLPSL language code that defines the configuration of the sessions, the environment and security goals to be achieved by our proposed scheme. The figures also show the definitions of the security goals declared to be secrets in the entity’s functions and the values that are authenticated by the entities.

Fig. 11 shows the protocol execution using SPAN software, where the all agents exchange the messages in authentication phase.

In conclusion, the results shown in Figs. 12 and 13 clearly show that the proposed scheme is immune against man-in-the-middle and replay attacks.
role role_S(S,M,G,D:agent,H:hash_func,SND,RCV:channel (dy))
played_by S
def=
  local;
  State:nat, IDm, IDs, N1, N2, N3: text, V1: hash
  (text.text), Vm, Vg: hash(text.message),
  M5, SG, GD, GS, SM, M1, M2, DIDs: message, Hms, Hsg, Hgs, M
  (message)
  init
  State := 1
  transition
  1. State=1 \ RCV(MS,Hms) =|> State':=3 \ M2'::=DIDs.N1.N2) / \ SG'::=xor(M2,Vg) / \ Hsg'::=H(M2) / \ SND(SG,Hsg)
  3. State=3 \ RCV(GS,Hgs) =|> State':=5 / \ M2'::=N1.N2.N3) / \ SM'::=xor(M2,Vg) / \ Hsm'::=H(M2) / \ SND(SM,Hsm)
end role

Fig. 7. Role of S in HLPSL code.

role role_M(M,S,G,D:agent,H:hash_func,SND,RCV:channel (dy))
played_by M
def=
  local;
  State:nat, IDm, N1, N2, N3, Lm: text, V1: hash
  (text.text), Vm: hash
  (text.message), DIDs, SM, M1: message, Hms, Hsm: hash
  (message), SK: hash(text.text)
  init
  State := 0
  transition
  0. State=0 \ RCV(start) =|> State':=2 \ DIm'::=DIDs.IDm.N1) / \ M1'::=DIDm.N1.Lm) / \ MS'::=xor
  (M1, Vm) / \ Hms'::=H(M1) / \ SND(MS, Hms) / \ secret
  (IDm, sec_idm, {S, M, G, D}) / \ secret(N1, sec_n1,
  {S, M, G, D})
  2. State=2 \ RCV(SM', Hsm') =|> State':=4
end role

Fig. 8. Role of M in HLPSL code.

role role_G(M,S,G,D:agent,H:hash_func,SND,RCV:channel (dy))
played_by G
def=
  local;
  State:nat, IDm, N1, N2, N3: text, V2: hash
  (text.message), V3: hash
  (message), SG, GD, DG, GS, M1, M2, M3, M4, M7, DIDs: message,]
  (message), Vg, Vd: hash(text.message), SK: hash
  (text.text)
  init
  State := 30
  transition
  30. State=30 \ RCV(GS', Hsg') =|> State':=32 \ M3'::=N1.N2.N3) / \ GD'::=xor(M3, Vd) / \ Hgd'::=H(M3) / \ SND(GD, Hgd)
  32. State=32 \ RCV(GD', Hgd') =|> State':=34 \ M3'::=N1.N2.N3) / \ GS'::=xor(M3, Vd) / \ Hgs'::=H(M3) / \ SND(GS, Hgs)
end role

Fig. 9. Role of G in HLPSL code.
5.4. Performance comparison

In this section, we assess the performance of our proposed scheme in terms of storage cost, computation costs, and communication overhead.

5.4.1. Storage cost

We evaluate storage cost (in bits) for M, S, G and D of the four participants.

M stores $I_d^m$ and $V_m$. As an example of hash function we utilize SHA3-384, and output of SHA3-384 is 384 bits. Using SHA3-384 we get $V_m = 384$ bits [56]. While $I_d^m = 128$ bits. Thus, the total storage required by M is $384 + 128 = 512$ bits.

S stores $I_d^s$, $I_d^m$, $I_d^g$, $V_m$ and $V_g$. By applying SHA3-384, we obtain $V_m = V_g = 384$ bits. The $I_d^s = I_d^m = I_d^g$ = 128 bits. Therefore, the total storage required by S is $(2x384) + (3x128) = 1152$ bits.

G stores $I_d^g$, $I_d^s$, $I_d^d$, $V_g$ and $V_d$. The $I_d^g = I_d^s = I_d^d = 128$ bits and $V_g = V_d = 384$ bits. Therefore, the total storage required by G is $(3x128) + (2x384) = 1152$ bits.
Table 4 shows the time required to conduct certain operations. However, the execution time of the bitwise XOR operation is negligible. Our scheme performs 13 hash invocations and 19 XOR operations, which yields a total computation cost \((13 \times T_h)\). Hence, the computation cost of our proposed protocol is \((13 \times 0.5 \text{ ms}) = 6.5 \text{ ms}\).
Table 5
Comparison of computation cost between the proposed scheme and other most related schemes in ms.

<table>
<thead>
<tr>
<th>Authentication scheme</th>
<th>Total cost</th>
<th>Rough estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kumar et al. [16]</td>
<td>$8T_1 + 10T_H$</td>
<td>74.6 ms</td>
</tr>
<tr>
<td>Chen et al. [22]</td>
<td>$2T_{sign} + 2T_R + 6T_H + 4T_H$</td>
<td>841.8 ms</td>
</tr>
<tr>
<td>Karthigaiveni et al. [58]</td>
<td>$2T_A + 3T_H + 2T_H$</td>
<td>630.3 ms</td>
</tr>
<tr>
<td>Alzahrani et al. [59]</td>
<td>$17T_H + 5T_C$</td>
<td>52 ms</td>
</tr>
<tr>
<td>Proposed scheme</td>
<td>$13T_A$</td>
<td>6.5 ms</td>
</tr>
</tbody>
</table>

Table 6
Security and functionality features comparison.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutual authentication</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Session key agreement</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>User anonymity</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Untraceability</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Forward security</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Password guessing attack</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Mobile device loss attack</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Privileged insider attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Impersonation attack</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Replay attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Man-in-the-middle attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Password change phase</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Formal proof (BAN logic)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Formal verification (AVISPA)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Authentication based on contextual factors</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Cloning attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Physical attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

A comparison of the computation cost between the proposed scheme and the other most related schemes in ms is shown in Table 5.

Table 6 provides security and functionality features compared to other existing related schemes.

6. Conclusion

Basic security requirements such as privacy, authentication, integrity, non-repudiation and key exchange are essential for telehealth delivery to remote communities and at-home patients. In this work a robust lightweight authentication and key exchange scheme is proposed among four entities involved in an telehealth system. After a round of secure authentication between a mobile device, server, gateway and an IoT edge device, a symmetric session key is established. Security of the proposed scheme is proved formally using BAN logic. In addition, informal security verification assures resistance of the proposed scheme in the face of the most common attacks. Finally, formal security of the proposed protocol is evaluated using the AVISPA tool that assures us of the security of the protocol.

Declaration of Competing Interest

The authors certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers’ bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

References

M. Fakroon, F. Gebali and M. Mamun
Internet of Things 13 (2021) 100343


