Journal Pre-proof

ECCbAS: An ECC based authentication scheme for healthcare IoT systems

Mohammad Reza Servati, Masoumeh Safkhani

PII: S1574-1192(23)00011-1
DOI: https://doi.org/10.1016/j.pmcj.2023.101753
Reference: PMCJ 101753

To appear in: *Pervasive and Mobile Computing*

Received date : 11 July 2022
Revised date : 7 November 2022
Accepted date : 18 January 2023

Please cite this article as: M.R. Servati and M. Safkhani, ECCbAS: An ECC based authentication scheme for healthcare IoT systems, *Pervasive and Mobile Computing* (2023), doi: https://doi.org/10.1016/j.pmcj.2023.101753.

ECCbAS: An ECC based authentication scheme for healthcare IoT systems

Mohammad Reza Servati, Masoumeh Safkhani

Department of Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran, Postal code: 16788-15811

Department of Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran, Postal code: 16788-15811 and School of Computer Science, Institute for Research in Fundamental Sciences (IPM), P. O. Box 19395-5746, Tehran, Iran.

ARTICLE INFO

Keywords:
Authentication protocol
MWSN
Integrity Contradiction Attack
Traceability Attack
Secret Disclosure Attack
de-synchronization Attack
Elliptic Curve Cryptography

ABSTRACT

Smart technology is a concept for efficiently managing smart things such as vehicles, buildings, home appliances, healthcare systems and others, through the use of networks and the Internet. Smart architecture makes use of technologies such as the Internet of Things (IoT), fog computing, and cloud computing. The Smart Medical System (SMS), which is focused on communication networking and sensor devices, is one of the applications used in this architecture. In a smart medical system, a doctor uses cloud-based applications such as mobile devices, wireless body area networks, and other cloud-based apps to provide online therapy to patients. Consequently, with the advancement and growth of IoT and 6G wireless technology, privacy and security have emerged as two of the world's most important issues. Recently, Sureshkumar *et al.* proposed an authentication scheme for medical wireless sensor networks (MWSN) by using an Elliptic Curve Cryptography (ECC) based lightweight authentication protocol and claimed that it provides better security for smart healthcare systems. This paper will demonstrate that this protocol is susceptible to attacks such as traceability, integrity contradiction, and de-synchronization with the complexity of one run of the protocol and a success probability of one. Furthermore, we also propose an ECC based authentication scheme called ECCbAS to address the Sureshkumar *et al.* protocol's vulnerabilities and demonstrate its security using a variety of non-formal and formal methods.

1. Introduction

The Internet of Things is a network of physical devices, such as cars, items, homes, and other things that are embedded with software, electronics, sensors, and network connectivity Atzori et al. (2010). These devices are linked together and exchange data with one another and other digital devices without the need for human intervention Al-Fuqaha et al. (2015) and Kouicem et al. (2018). Smart cities, e-healthcare, smart homes, smart grids, and other Internet of Things applications all help to improve our quality of life due to the rapid development of IoT, Internet communication with embedded applications for information sharing in recent years. The embedded device has limited capacity, power, and computational capabilities. As a result, it is linked to a cloud server, which has higher storage capacity and power and can also handle most IoT concerns. One of the uses of the Internet of Things, as previously said, is in e-healthcare.
There are numerous definitions for telemedicine, but according to the World Health Organization (WHO), telemedicine originally referred to the delivery of health care over a large geographic area using advanced telecommunication capabilities and the Internet. Such healthcare services can also include regular checkups for patients, storing, analysis, transfer of scans or photos, patient therapy and discussion via multimedia, patient tele monitoring, and even doctors' consulting from clinicians all over the world Jin and Chen (2015). In this type of application, there is a connection between the doctor and the patient, as well as the hospital and the patient's house. The doctor and the patient can communicate with each other through this system. This communication will take place over a communication channel, and the information exchange will be handled by a server. The channel of communication must be secure. Also, the patient's trust is required for data entry to the cloud server, which is collected and sent via a secure channel to the cloud server, whereas data transmission over an insecure channel poses a challenge Stergiou et al. (2018).
Figure 1 displays research challenges in IoT, one of which is the increased use of sensing and IoT devices in healthcare systems, such as fingerprint scanners, thermometers, and other devices. In addition, privacy and security based on Figure 1 play an important role in the hospital on its own storing a patient's medical information in hard copy or soft

Mohammadreza.servati@sru.ac.ir (M. Safkhani); Safkhani@sru.ac.ir (M. Safkhani)
ORCID(s):

copy form. The most serious problem with this type of medical file storage is the possibility of data theft, which could be used for illegal clinical and academic studies or by insurance providers. This unauthorized access to medical data has a significant impact on a patient's privacy. As a result, protecting a person's medical information is the primary challenge, according to Gai et al. (2018).

The medical community has been able to better manage patients through the use of sensing devices thanks to the development of wireless body area networks (WBAB) and the expansion of social life. In terms of cost savings, this is one benefit for the patient's treatment and is a simple way to check blood glucose levels, heartbeat, cholesterol levels, and other patient status indicators Abidi et al. (2017).

Besides, the authentication techniques that have been investigated are classified into the following categories: Mutual, one-time password (OTP), SecureID, group, two-party through a trusted party with key exchange, session key-based authentication, and directed path-based authentication. Authentication protocols are also one of the most common and simple methods for securing network-based applications Hammi et al. (2020). These protocols are commonly used in a number of settings, including multi-server environments Li et al. (2013) and Li et al. (2012) and Li et al. (2015), RFID systems Niu et al. (2014), and satellite communication systems Lee et al. (2012). It is also an important mechanism for protecting WSNs from security attacks, as a result, some effort is being made to develop authentication protocols for use on WSNs and WMSNs. In terms of privacy preservation, this is one advantage for the patient. Since secure transmission of data to a cloud server is crucial, various security attacks such as eavesdropping, impersonation attacks, replay attacks, and so on are possible Gupta (2018) and Gupta et al. (2016).

The majority of machine learning (ML) techniques used in the healthcare system concentrate on the assessment of electronic health records (EHR) data, which may include meetings, diagnostics, remedies, and exams like medical imaging Keyhani et al. (2008). EHR developed as a result of medical institutions' natural use of technology and became a useful source of information for many analytical techniques intended to extract knowledge to support medical practices King et al. (2014).

A learning model that has been trained to recognize patterns and extract the desired knowledge from raw data is necessary for ML-based methods. This training step is essential for the resulting analysis method to be accurate. To ensure the quality of the ML model, a large dataset with a diverse range of samples must be used Chilimbi et al. (2014). The quantity of data points, the variety of samples, and the caliber of dataset annotation with regard to the anticipated classification all have a direct impact on the training's outcomes. Such datasets are often time-and expense-consuming to acquire or produce. In this process, data is gathered from various sources, moved to a central data repository, and then combined to create a model. The main obstacles to the development of cutting-edge ML techniques for the healthcare industry are these challenges and limitations on the sharing of medical data Horvitz and Mulligan (2015).

Federated learning (FL) is one of the approaches to solving these problems. FL enables the training of ML models domestically (at the location of the data) and only gives the requesting party access to the final model, which cannot be reverse-engineered Yang et al. (2019). By preventing exposure to organizations conducting studies and enabling data usage for greater purposes, FL eliminates the need to share sensitive information and private datasets with others. Each participating data holder receives the training algorithm from a central organization, which also controls the learning process. With the help of their own personal data, each participant creates a local model that they then train, sharing the output parameters with the central entity. To create a single global model, the central entity eventually uses an aggregation algorithm to merge the parameters of all local models Antunes et al. (2022). For instance, papers such as [Bellavista et al. (2021), Rieke et al. (2020), Zerka et al. (2020), Li et al. (2020)] deal with the healthcare system. Figure 2 depicts the typical infrastructure of FL in healthcare systems.

Furthermore, Blockchain is a new technology that Satoshi Nakamoto invented in 2008, and many researchers are interested in using it for many aspects, such as decentralization, consistency, validation, and etc. These features play significant roles in privacy, monitoring patients, and data sharing between entities such as hospitals, doctors, nurses, and others Narwal and Mohapatra (2021) which can employed in the building of smart and secure healthcare system. As a result, medical systems can benefit from our proposed approach for real-world implementation to create a standard foundation for smart medical systems. According to the literature analysis, there is still room for development in terms of security attacks. The majority of the protocols proposed in the literature contain security flaws, such as a lack of user privacy. Furthermore, most of them are susceptible to a certain security threat. Generally, in order to provide communication security in medical wireless sensor networks, the security protocols must be fully safe. In addition, RFID authentication algorithms based on Elliptic Curve Cryptography (ECC) have been utilized to successfully address security and privacy concerns in IoT application domains. In this paper, we examine the Sureshkumar et al. (2019)'s authentication protocol and show that this protocol does not have the necessary resistance to attacks. We also propose

Figure 1: Research challenges in IoT

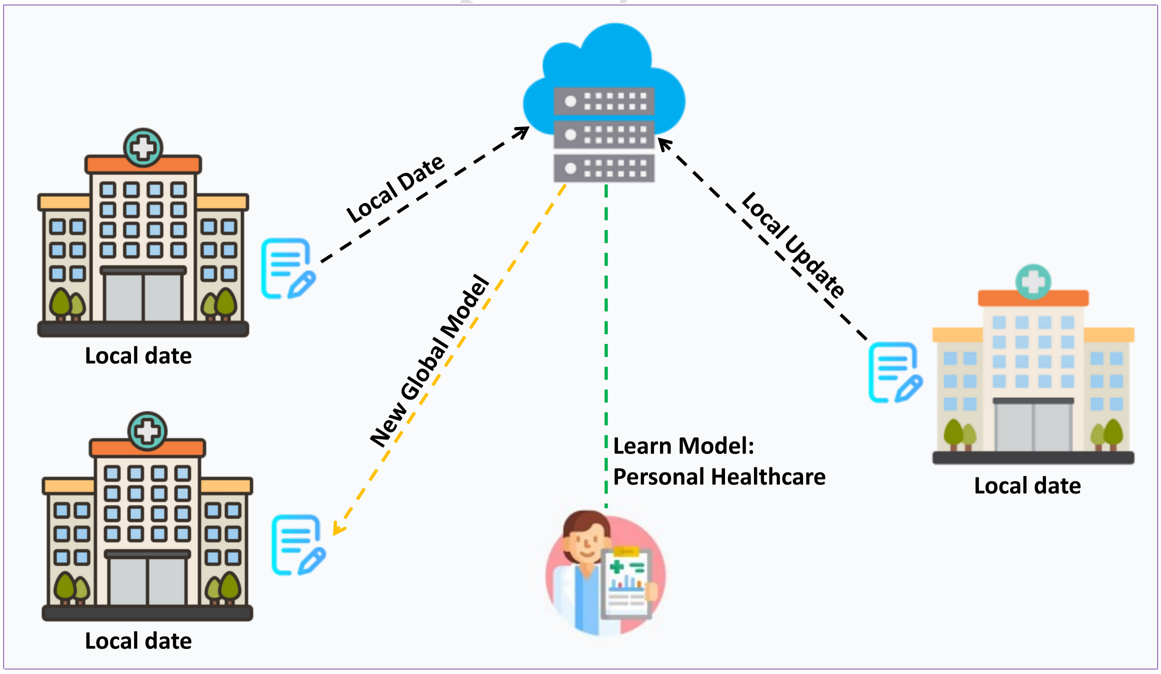


Figure 2: A typical federated learning (FL) infrastructure which is applied in healthcare systems

an improved version of it called ECCbAS. The proposed scheme includes two channels: one is for transmitting data during the enrollment phase and one for communications during the authentication and password change phases.

1.1. Main Contribution

The following are this paper's contributions:

1. This paper shows that the proposed scheme by Sureshkumar et al. (2019) is prone to a multitude of attacks such as traceability, de-synchronization and integrity contradiction.
2. We addressed Sureshkumar et al. (2019)' security flaws and proposed an improved protocol named ECCbAS. The security of ECCbAS is also proved through both informal and formal security analysis. The comparisons of ECCbAS with similar recent protocols show that our suggested protocol has better security and performance than its predecessor, i.e., the Sureshkumar et al. (2019)' protocol.
3. In this paper, we divide our security analysis into formal and informal methods. Scyther and ProVerif are the formal automatic tools used, and BAN logic is the manual formal method used.

1.2. Paper Organization

The rest of this paper is structured as follows: In Section 2, the related work in this field is briefly reviewed. The healthcare authentication protocol using the cloud, which was proposed by Sureshkumar *et al.* is discussed in Section 3. Section 4 declares the security vulnerabilities of the Sureshkumar *et al.*'s protocol, including traceability, integrity contradiction, and de-synchronization attacks. The proposed scheme to address the security pitfalls of Sureshkumar *et al.*'s protocol for healthcare systems called ECCbAS is described in Section 5. Section 6 and Section 7, respectively, explain security analysis and performance analysis, as well as security, storage, computational, and communication comparisons of ECCbAS with other similar protocols. Finally, Section 8 concludes this paper with concluding remarks.

2. Related Work

Many authentication protocols were investigated over the years, but they had no adequate or reasonable security properties against a wide range of attacks. In this section, we looked at some of them. For instance, Xue et al. (2013) proposed one authentication scheme for wireless medical networks, which later Jiang et al. (2015) demonstrated their protocol is not secure and is suspicious to off-line password guessing and traceability attacks. Although, Das (2016) showed Jiang et al. (2015) protocol is vulnerable to user forgery and de-synchronization attacks.

Wu et al. (2017) suggested a scheme for wireless medical sensor networks and claimed that their protocol is safe, but Srinivas et al. (2017) showed that their protocol is vulnerable to a variety of attacks, such as insider and off-line password guessing attacks. However, Srinivas *et al.*'s protocol does not withstand off-line guessing attacks.

Jia et al. (2019) suggested a biometric-based authentication scheme for the e-healthcare system in a fog server environment. Salem and Amin (2020) provided an RFID protocol for telecare medicine information systems (TMIS) based on ElGamal cryptography and verified its security with the AVISPA tool Armando et al. (2005).

Kumar et al. (2020) proposed an RFID-based mutual authentication and key agreement protocol for vehicular cloud computing and claimed that it is secure, but Safkhani et al. (2021) stated that their protocol is unreliable and vulnerable to impersonation and relay attacks. Kumari et al. (2014) proposed a key agreement-based smart card-based remote user authentication strategy, and claimed that their scheme is appropriate, secure, and efficient for real-world applications. However, Kaul and Awasthi (2016) illustrated that, this protocol is wholly insecure, because an adversary could obtain entry not just to the protocol's security mechanisms, but also to the popular authenticator for future interaction between the user and the server. They also showed that in this protocol, an attacker also obtains the enrolled user's password and the server's decryption key. Also, Kaul and Awasthi (2016) proposed new protocols to stay safe. However, Rana et al. (2021) proved that the Kaual and Awasti protocols are insecure in such a way that an attacker can easily discover the identity of a legit user sending data over the public channel. Furthermore, an attacker can spoof a valid user of the system to reap the benefits of the server's services and use the authenticity of a genuine user. As a consequence, the Kaual and Awasti protocols are vulnerable to spoofing attacks, and their assertion of security is debunked.

Arshad and Rasoolzadegan (2016) proposed a scheme for Telecare Medicine Information Systems with User Privacy Protection, but Ostad-Sharif et al. (2019) demonstrated that their protocol is not secure and vulnerable to key compromise impersonation attack. He et al. (2017) suggested a Wireless Body Area Network with Anonymous Authentication and Provable Security while Sowjanya et al. (2021) shows that their proposed protocol is not withstanding clock synchronization and insider attacks. In other schemes, the authentication technique provided by Das et al. (2019) was enhanced, and after that, Chaudhry et al. (2020) discovered that their protocol was susceptible to device impersonation and man-in-the-middle attacks. Furthermore, Ali et al. (2020) demonstrated that the protocol of Challa et al. (2018) are vulnerable to inaccuracy, broadcasting issues, a lack of sensor node and Trusted Authority (TA) authentication, a replay attack, a DoS attack, a forgery attack, and a communication delay. Also, Challa et al. (2018) illustrated that the Liu and Chung (2017)'s protocol is vulnerable to stolen smart-card, offline password guessing, privileged insider, and user impersonation attacks. Ali et al. (2020) identified that the protocol of Liu and Chung (2017) is not strong either against users' private key leakage and user impersonation attacks towards sensors.

Arslan et al. (2021) published an authentication protocol for real-world use. Also, Arslan and Bingöl (2022) published another scheme and demonstrated that Gabsi et al. (2021)'s scheme is not secure against traceability attacks, tag anonymity contradiction attacks, and forward and backward security contradiction attacks. Rostampour et al. (2022) proposed another protocol for resource-limited environments such as the Internet of Things. Wei et al. (2022) demonstrated that Qian et al. (2016)'s scheme is insecure and vulnerable to impersonation attacks. Kumar et al. (2022) proposed another mechanism for RFID networks based on the Internet of Things. As another scheme Mubarakali (2021) proposed a Blockchain based authentication scheme for wireless sensor networks.

Sureshkumar et al. (2019) recently proposed a lightweight authentication protocol based on Elliptic Curve Cryptography (ECC) and claimed that it provides better security for smart healthcare systems. While this paper shows that, unfortunately, this protocol is vulnerable to attacks such as traceability, integrity contradiction, and de-synchronization. The complexity of all presented attacks is equal to one run of the protocol and also that their success probability is equal to one. Furthermore, we propose an ECC-based authentication scheme called ECCbAS to address the vulnerabilities of the Sureshkumar *et al.* protocol and demonstrate its security using a variety of non-formal and formal methods. In addition, Table 1 shows a summary of related work.

3. Protocol Review

In this section, we describe the Sureshkumar *et al.*'s authentication scheme, including the System Model, Gateway and Sensor Nodes Registration Phase, User Registration Phase, Login Phase, Authentication Phase, and Password Update Phase. Table 3 shows the notations used throughout the paper.

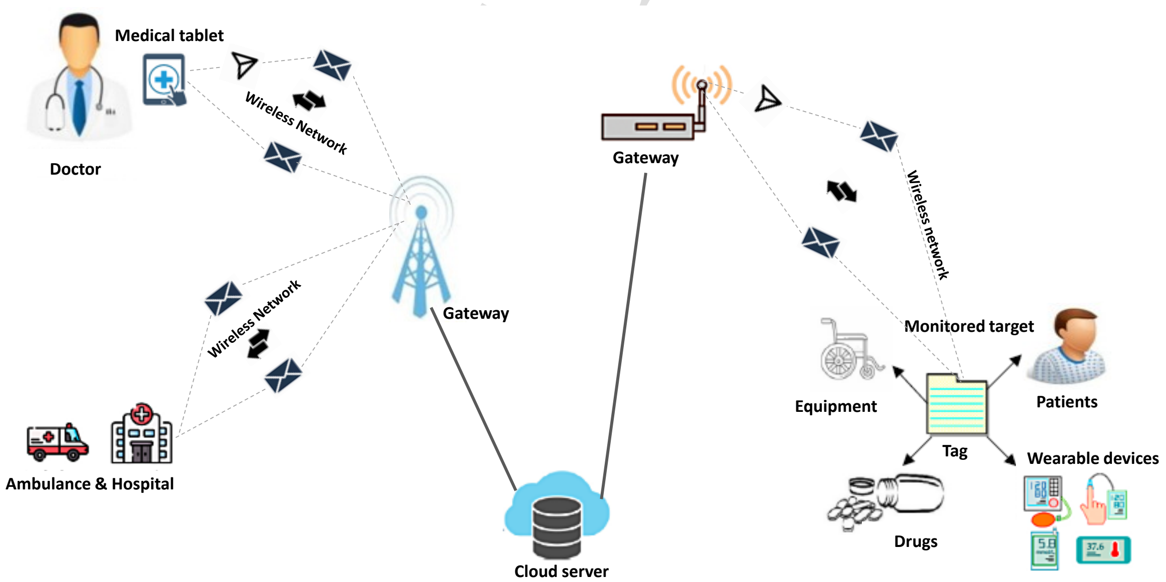


Figure 3: Infrastructure of a typical wireless medical sensor network

Table 1
A summary of related works

References	vulnerabilities	The authors who published vulnerabilities.
Xue et al. (2013)	1) Off-line password guessing attack 2) Traceability attacks	Jiang et al. (2015)
Jiang et al. (2015)	1) Forgery attacks 2) De-synchronization attacks	Das (2016)
Wu et al. (2017)	1) Insider attacks and off-line attacks 2) Password guessing attacks	Srinivas et al. (2017)
Jia et al. (2019)	-	-
Salem and Amin (2020)	-	-
Kumar et al. (2020)	1) Impersonation attacks 2) Relay attacks	Safkhani et al. (2021)
Safkhani et al. (2021)	-	-
Kumari et al. (2014)	1) Obtaining password 2) Obtaining server decryption key	Kaul and Awasthi (2016)
Kaul and Awasthi (2016)	Spoofing attacks	Rana et al. (2021)
Rana et al. (2021)	-	-
Sureshkumar et al. (2019)	1) Traceability attacks 2) Integrity contradiction attacks 3) De-synchronization attacks	Our self in this paper
Arshad and Rasoolzadegan (2016)	Impersonation attacks	Ostad-Sharif et al. (2019)
He et al. (2017)	1) Clock de-synchronization attacks 2) Insider attacks	Sowjanya et al. (2021)
Sowjanya et al. (2021)	-	-
Das et al. (2019)	1) Device impersonation attacks 2) Man-in-the-middle attacks	Chaudhry et al. (2020)
Chaudhry et al. (2020)	-	-
Challa et al. (2018)	1) Replay attacks 2) DoS attacks 3) Forgery attacks	Ali et al. (2020)
Liu and Chung (2017)	1) Impersonation attacks 2) Stolen smart card attacks 3) Offline password guessing attacks 4) etc.	Challa et al. (2018) Ali et al. (2020)
Ali et al. (2020)	-	-
Arslan et al. (2021)	-	-
Gabsi et al. (2021)	1) Tag anonymity contradiction attacks/ traceability attacks 2) Forward and backward security	Arslan and Bingöl (2022)
Arslan and Bingöl (2022)	-	-
Rostampour et al. (2022)	-	-
Kumar et al. (2022)	-	-
Qian et al. (2016)	Impersonation attacks	Wei et al. (2022)
Wei et al. (2022)	-	-
Mubarakali (2021)	-	-

3.1. System Model

Sureshkumar *et al.* proposed a healthcare system model, which is depicted in Figure 3. In this model, smart sensors such as blood pressure, blood glucose level, and body temperature collect data and send it to the GW_j, where the data is updated on a regular basis. The patient's data is sent through sensors on his body via wireless communication channels; this wireless communication can be Bluetooth, Zigbee, or infrared technologies. While the patient is in the hospital, a doctor in the hospital can connect to the gateway and examine the patient's condition, and the doctor can obtain data from the patient anywhere. Despite the fact that the patient is in the hospital and his or her data is saved

Table 2
Notations

Symbol	Description
SA	System Administrator
GW_{ID}	Gateway Identifier
U_i	User
SN_K	Sensor node
SC	Smart card
GW_j	Gateway node
S_{GW_j}	A secret value of gateway node
S_{SN_k}	A secret value of sensor node
ECC	Ecliptic Curve Cryptography
S_{SA}	Secret key of system administrator
n	Corresponds to $n = p.q$
r_u, r_s, r_g	Random numbers generated by user, sensor and gateway respectively
M	Message
F_q	A finite field consisting of q elements $\{0, 1, ..., q-1\}$, where q is a prime number
ΔT	Time threshold
F_q^*	The collection of integers number $\{1, ..., q-1\}$, i.e. $F_q^*/0$
h	One-way hash function
$A \stackrel{?}{=} B$	Check to see if A and B are equal or not
$\oplus$	Bitwise Exclusive-OR operation
$\|$	Concatenation operation
sk	Secret key

on a cloud server, this communication is not secure and should be established prior to it through user authentication.

3.2. Initialization Phase

During this step, the system administrator (SA) manually configures each entity on the cloud server, and each entity has its own credential stored on the server. For use in smart devices, this protocol employs lightweight elliptic curve cryptography (ECC): $E(F_q) = \langle p, q, a, b, n, G(P) \rangle$ with the long term secret key $S_{SA} \in F_q$.

3.3. Gateway and sensor node registration phase

The gateways (GW) and sensor nodes (SN) are entities that must be manually enrolled. The System Administrator (SA) accomplishes this phase by following the steps articulated below to enroll them.

1. SA chooses a GW_{ID_j} identity for GW_j, after that it computes and stores the value $S_{GW_j} = h(S_{SA} \| GW_{ID_j})$ in its database. Furthermore, SA stores $\langle GW_{ID_j}, S_{GW_j} \rangle$ in the memory of the gateway GW_j.

2. The SA determines an identity SN_{ID_K} for the K^{th} sensor node and calculates $S_{SN_K} = h(S_{SA} \| SN_{ID_k})$ then keeps $\langle SN_{ID_k}, S_{SN_k} \rangle$ and also stores this value in both sensor node and gateway node. The SA also stores GW_{ID_j} in the memory of sensor node. The sensor node and gateway registration method are expressed in Algorithm 1.

It is worth noting that during user login and authentication phases, the GW_j and SN_k are authenticated using these shared secret credentials. Finally, the identities of registered gateway nodes are published by SA for user access. Also, the identities of sensor nodes are kept secret in order to maintain their confidentiality.

Data: Information to be stored in entities memories.
Result: Registering sensors and gateway nodes

1: SA chooses identity GW_{ID_j} for GW_j.
2: SA computes $S_{GW_j} = h(S_{SA} \| GW_{ID_j})$ and saves it in its database in touch to GW_j.
3: SA chooses an identity SN_{ID_K} for the K^{th} sensor node SN_K.
4: SA computes $S_{SN_k} = h(S_{SA} \| SN_{ID_K})$.
5: SA saves $\langle SN_{ID_k}, S_{SN_K} \rangle$ in the memory of both GW_j and SN_K.
6: The SA also stores GW_{ID_j} in the memory of sensor node.

Algorithm 1: Sureshkumar *et al.*'s procedure for sensors and gateway nodes registration

Data: Personality information such as $\langle ID_i, PW_i, B_i \rangle$
Result: Users Registration

1: U_i selects its ID_i and PW_i
2: U_i calculates $b_i = H(B_i)$
3: U_i calculates $HID_i = h(ID_i \| b_i)$ and $HPW_i = h(PW_i \| b_i)$
4: U_i sends $\langle HID_i, HPW_i, GW_{ID_j} \rangle$ to SA
5: SA calculates $A_2 = h(HID_i \| HPW_i).P$,$A_2 = h(HID_i \| S_{GW_j}).P$, $A_3 = A_2 \oplus A_1$ and $A_4 = S_{GW_j}.P$
6: SA creates smart card SC=$\langle A_3, A_4, h(\cdot), P \rangle$
7: SA sends SC to U_i.

Algorithm 2: The user registration procedure in the Sureshkumar *et al.*'s protocol

3.4. User Registration Phase

A reputable user is granted access to the detected data after successful user authentication. This event occurs when the gateway node reads the sensor's observed data and wants to send the data to the user. Therefore, we need a procedure to enroll users. As a result, the process of user registration is explained further below.

1. The user U_i selects ID_i and PW_i and computes its bio hashing $b_i = H(B_i)$ with its identity ID_i and password PW_i. After that, U_i computes $HID_i = h(ID_i \| b_i)$ and $HPW_i = h(PW_i \| b_i)$ and then sends $\langle HID_i, HPW_i, GW_{ID_j} \rangle$ to SA.
2. Once received the message, SA calculates $A_1 = h(HID_i \| HPW_i).P$,$A_2 = h(HID_i \| S_{GW_j}).P$, $A_3 = A_2 \oplus A_1$, $A_4 = S_{GW_j}.P$ and then creates the $SC = \langle A_3, A_4, h(.), P \rangle$ and sends it to the user through a secure channel.
3. After receiving the SC from the SA, the user computes: $HPID_i = h(HID_i \| PW_i)$, $A_5 = h(HID_i \| HPID_i).P$,$A_2^* = A_3 \oplus A_1$ and $A_6 = A_2^* \oplus A_4$. Now the user smart card is SC=$\langle A_3$,A_5,A_6,$h(.)$, $P \rangle$ after replacing A_4 with A_6 in the smart card and storing A_5.

Algorithms 2 shows the summary of the user registration procedure of Sureshkumar*et al.*'s protocol.

3.5. Login phase

To access data, the user must login via the gateway node GW_j. The steps that follow demonstrate how the user logs into the GW_j.

1. The U_i inserts SC into card reader or terminal, then enters its identity ID_i and password PW_i with biometric B_i.
2. The smart card computes: $b_i = h(B_i)$, $HID_i = h(ID_i \| b_i)$, $HPID_i = h(HID_i \| PW_i)$ and $A_5^* = h(HID_i \| HPID_i).P$.
3. The SC determines whether $A_5^* \stackrel{?}{=} A_5$. If this equality is incorrect, the protocol is terminated; otherwise, SC selects a random number $r_u \in F_q$ and computes: $HPW_i = h(PW_i \| b_i)$, $A_1^* = h(HID_i \| HPW_i).P$, $A_2^* = A_3 \oplus A_1^*$, $A_7 = h(A_2^* \| T_1)$, $A_8 = r_u.P$, $A_9 = A_8 \oplus A_2^*$ and $A_{10} = A_6 \oplus A_9 = A_4 \oplus A_8$, where in A_7, T_1 shows timestamp.

4. The SC sends login messages $M_1 = \langle A_7, A_9, A_{10}, T_1 \rangle$ to the gateway node GW_j. Users can login into the system and access patient information directly using the sensor.

In addition, Algorithms 3 shows the summary of the user login procedure of Sureshkumar *et al.*'s protocol.

Data: U_i Enters his or her own SC into the terminal, along with personal information such as $\langle ID_i, PW_i, B_i \rangle$
Result: User login

1. U_i inserts his/her own SC and enters its ID_i, PW_i, and B_i.
2. SC computes $b_i, HID_i, HPID_i, A_5^*$.
3. SC checks whether $A_5^* \stackrel{?}{=} A_5$
   If this equality is incorrect, SC will terminate the protocol; otherwise, SC selects r_u and calculates HPW_i, A_1^*, A_2^*,A_7, A_8, A_9 and A_{10}.
4. At the end, SC transmits $M_1 = \langle A_7, A_9, A_{10}, T_1 \rangle$ to the GW_j.

Algorithm 3: The user login procedure in the Sureshkumar *et al.*'s protocol

3.6. Authentication phase

This step's goal is to authenticate the protocol entities and generate a shared secret key between the user, the gateway, and the sensor node. The steps below illustrate how to do so:

1. $U_i \rightarrow GW_j$
   After receiving login message request, the gateway node GW_j calculates the delay $\Delta T = T_2 - T_1$ by using the gateway node's timestamp T_2, and if the time delay ΔT is not acceptable, the login and authentication procedure fails. If the time delay is acceptable, GW_j computes $A_4^* = S_{GW_j}.P$, $A_8^* = A_{10} \oplus A_4$,$A_2^{**} = A_9 \oplus A_8^*$ and $A_7^* = h(A_2^{**} \| T_1)$. Then it determines whether $A_7^* \stackrel{?}{=} A_7$. If so, GW_j chooses a random number $r_g \in F_q$ and calculates: $A_{11} = r_g.A_8^* = r_u.r_g.P$, $A_{12} = r_g.P$, $A_{13} = h(S_{SN_k}).A_{12}$, $A_{14} = h(GW_{ID_j} \| S_{SN_k}).P$, $A_{15} = h(A_{14} \| T_2)$, and $A_{16} = A_8^* \oplus A_{13}$. Then GW_j sends $M_2 = \langle A_{12}, A_{11}, A_{15}, A_{16}, T_2 \rangle$ to the sensor node.
2. $GW_j \rightarrow SN_k$
   When the sensor node SN_k receives M_2 from the gateway node, it uses its own timestamp to confirm the delay of $\Delta T = T_3 - T_2$, and if this delay is incorrect, the protocol is terminated. Otherwise the sensor node SN_k calculates $A_{14}^* = h(GW_{ID_j} \| S_{SN_k}).P$, $A_{15}^* = h(A_{14}^* \| T_2)$ and checks whether $A_{15}^* \stackrel{?}{=} A_{15}$, if this equality holds, SN_k chooses a random number $r_s \in F_q$ and computes: $A_{17} = r_s.A_{12}$, $A_{18} = h(A_{17} \| S_{SN_K} \| T_3)$, $A_{13}^* = h(S_{SN_k}).A_{12}$, $A_8^{**} = A_{16} \oplus A_{13}$, $A_{19} = r_s.A_8^{**}$, and $A_{20} = r_s.P$. Following this phase, the sensor node SN_k calculates the session key as $sk = r_s.A_{11}$ and also sends $M_3 = \langle A_{19}, A_{18}, A_{20}, T_3 \rangle$ to GW_j.
3. $SN_K \rightarrow GW_j$
   After receiving M_3 from the sensor node SN_k, the GW_j node verifies the delay $\Delta T = T_4 - T_3$ by its own timestamp T_4, and if this delay ΔT is not correct, the gateway node rejects the procedure. Otherwise, it computes $A_{17}^* = r_g.A_{20}$ and checks whether $A_{18}^* = h(A_{17}^* \| S_{SN_k} \| T_3) \stackrel{?}{=} A_{18}$. If so, then GW_j computes $A_{21} = h(A_2^{**} \| A_4^* \| A_8^*)$ and sends $M_4 = \langle A_{17}^*, A_{21} \rangle$ to the user. Furthermore, the gateway node computes the session key as $sk = r_g.A_{19}$.
4. $GW_j \rightarrow U_i$
   Following the receipt of message M_4 from the GW_j, the user computes $A_4^* = A_6 \oplus A_2^*$, $A_{21}^{**} = h(A_2^* \| A_8^* \| A_4^*)$ and checks equality whether $A_{21}^* \stackrel{?}{=} A_{21}$. If so, the U_i will calculate the session key as $sk = r_u.A_{17}^*$.

The process of Sureshkumar *et al.*'s login and authentication phase is also shown in Figure 4. Furthermore, Algorithms 4 shows the summary of the user authentication procedure of Sureshkumar *et al.*'s protocol.

Data: $M_1 = \langle A_7, A_9, A_{10}, T_1 \rangle$
Result: Authenticate the user

1. GW_j receives $M_1 = \langle A_7, A_9, A_{10}, T_1 \rangle$ from U_i.
2. After checking timestamp by GW_j, the GW_j computes A_4^*,A_8^*, A_2^{**} and A_7^*.
3. GW_j checks whether $A_7^* \stackrel{?}{=} A_7$
   If this equality is not sensible, GW_j will terminate the protocol; otherwise, GW_j selects r_g and calculatesA_{12}, A_{13}, A_{14}, A_{15} and A_{16}.
4. GW_j sends M_2 to SN_k.
5. After receiving M_2 from GW_j, SN_k checks timestamp and calculates A_{14}^* and A_{15}^*.
6. SN_K checks whether $A_{15}^* \stackrel{?}{=} A_{15}$
   If this equality was wrong, SN_k will terminate the protocol; otherwise, it chooses r_s and computes A_{17}, A_{18}, A_{13}^*, A_8^{**}, A_{19}, A_{20} and secret key (sk).
7. SN_k transmits M_3 to GW_j.
8. After getting M_3 from SN_k, GW_j checks timestamp and computes A_{17}^* and A_{18}^*.
9. GW_j checks whether $A_{18}^* \stackrel{?}{=} A_{18}$
   If this equality was not correct, SN_k will terminate the protocol; otherwise, it calculates A_{21}^* and secret key (sk).
10. GW_j sends M_4 to U_i.
11. After obtaining M_4 from GW_j, U_i checks timestamp and computes A_4^*.
12. U_i checks whether $A_{21}^* \stackrel{?}{=} A_{21}$
    If this equality was not true, U_i will terminate the protocol; otherwise, it computes secret key(sk).

Algorithm 4: The authentication procedure in the Sureshkumar *et al.*'s protocol

3.7. Update Password

To increase security, passwords must be changed periodically. For this purpose, the following steps are considered in the Sureshkumar *et al.*'s protocol:

1. U_i inputs his smart card into the terminal and enters his ID_i, PW_i with his biometric B_i.
2. A smart card (SC) calculates $b_i = H(B_i)$, $HID_i = h(ID_i \| b_i)$ and $HPID_i = h(HID_i \| PW_i)$. SC also calculates $A_5^* = h(HID_i \| HPID_i).P$ and checks whether $A_5^* \stackrel{?}{=} A_5$. When the equality is not true, SC terminates the session; otherwise, the SC allows U_i to enter a new password.
3. PW_i^{new} is the user's new password, which he enters into the SC.
4. The smart card calculates $HPID_i^{new} = h(HID_i \| PW_i^{new})$, $A_1^{new} = h(HID_i \| HPW_i^{new}).P$, $A_3^{new} = (A_3 \oplus A_1) \oplus A_1^{new}$, and $A_5^{new} = h(HID_i \| HPID_i^{new}).P$.

Finally, the SC replaces A_3 and A_5 with A_3^{new} and A_5^{new} respectively. Algorithm 5 shows how the password is updated in the Sureshkumar *et al.*'s protocol.

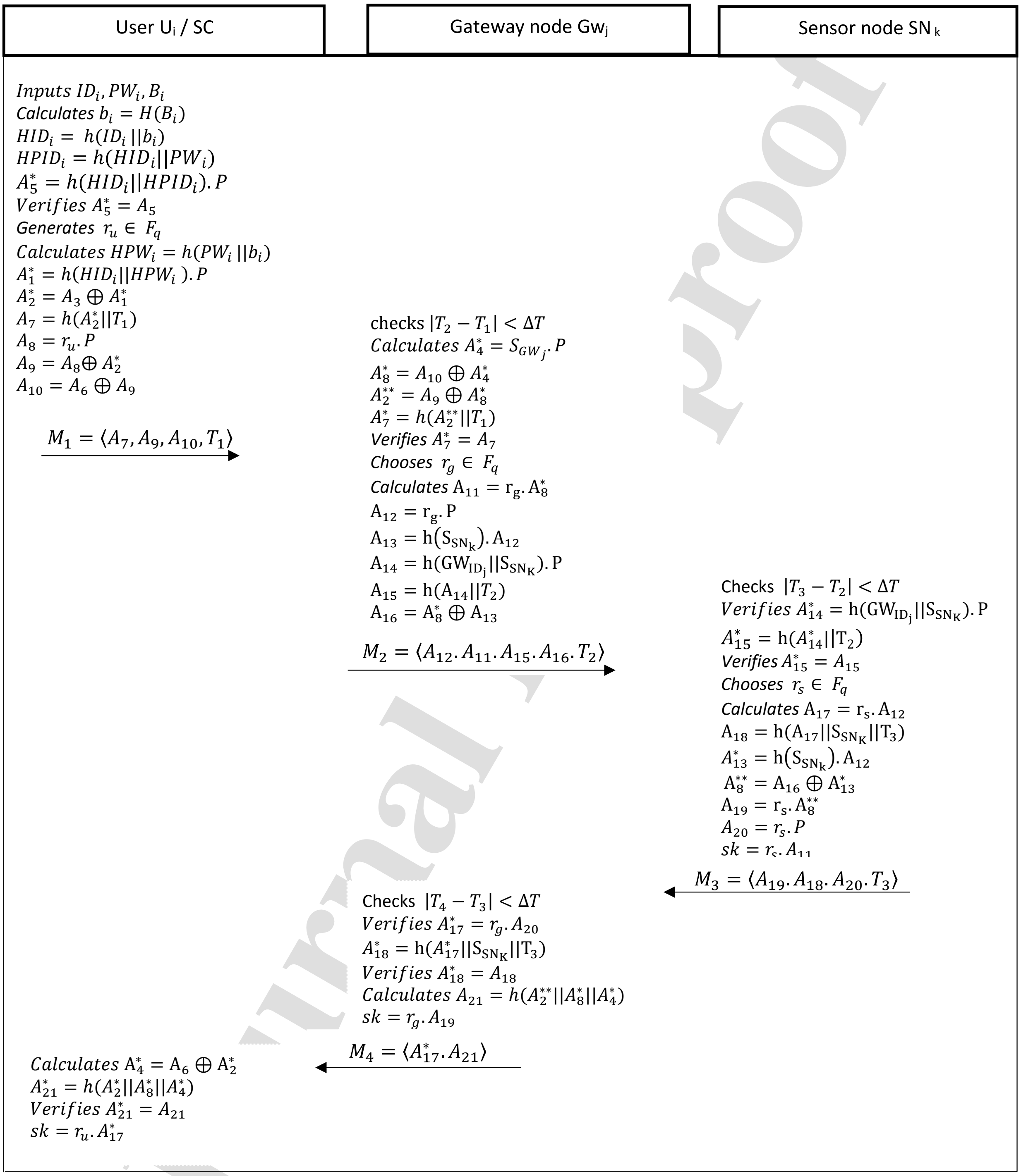


Figure 4: The login and authentication phase of Sureshkumar *et al.*'s protocol over a public channel

.

Data: Personality information such as $\langle ID_i, PW_i, B_i\rangle$
Result: Update password

1. U_i inserts his smart card and enters ID_i, PW_i, and his biometric B_i.
2. SC calculates $b_i = H(B_i)$, $HID_i = h(ID_i \| b_i)$ and $HPID_i = h(HID_i \| PW_i)$.
3. SC also calculates $A_5^* = h(HID_i \| HPID_i).P$
4. **if**$(A_5^* == A_5)$ **then**
   (a) U_i enters a new password using his SC.
   (b) U_i enters his new password PW_i^{new} into SC.
   (c) SC calculates $HPID_i^{new} = h(HID_i \| PW_i^{new})$, $A_1^{new} = h(HID_i \| HPW_i^{new}).P$, $A_3^{new} = (A_3 \oplus A_1) \oplus A_1^{new}$, $A_5^{new} = h(HID_i \| HPID_i^{new}).P$
   (d) SC substitutes old A_3 and A_5 with A_3^{new} and A_5^{new} respectively.
5. **else**
6. SC terminates the session.
7. **end if**

Algorithm 5: Algorithm of update password in Sureshkumar *et al.*'s protocol

3.8. Sensor Node Addition

When a sensor node is hacked by an attacker or loses its battery capacity, a new sensor node must be installed. The procedure for installing a new sensor node is outlined below.
SA chooses a new identity $SN_{ID_k}^{new}$ for the sensor node SN_K^{new} and calculates a secret value as follows: $S_{SN_k} = h(S_{SA} \| SN_{ID_k}^{new})$ and saves $\langle SN_{ID_k}, S_{SN_k}\rangle$ into the memory of both the gateway node GW_j and the sensor node SN_k.

4. Sureshkumar *et al.*'s Protocol Cryptanalysis

Sureshkumar et al. (2019) asserted that their scheme is safe and effective. However, as shown below, their protocol lacks the requisite resistance against traceability, integrity contradiction, and de-synchronization attacks.

4.1. Ttraceability attack

Patient anonymity and untraceability are critical considerations when developing an authentication protocol. If the patient's anonymity is compromised, the attacker can obtain personal sensitive information such as the medical record, movement patterns, social circle, and current location. Untraceability is also a security feature that ensures the server or adversary cannot determine which user is interacting with the gateway and sensor nodes. This concept can also be applied to gateways and sensors. Figure 5 also depicts an example of a traceability attack's concept. In the traceability attack against Sureshkumar *et al.*'s protocol, the attacker follows the steps below and will trace the user based on the information gathered from the exchanged messages in the protocol. The traceability attack against Sureshkumar *et al.*'s protocol is done as follows:

1. An adversary eavesdrops and saves the message $M_1 = \langle A_7, A_9, A_{10}, T_1\rangle$.
2. As mentioned in Sureshkumar *et al.* protocol, the value A_{10} is $A_{10} = A_9 \oplus A_6$, so $A_6 = A_{10} \oplus A_9$. On the other hand, according to phases of protocol A_6 is computed as $A_6 = A_2^* \oplus A_4$ where A_4 and A_2^* are respectively $A_4 = S_{GW_j}.P$, and$A_2^* = h(HID_i \| S_{GW_j}).P$. Therefore, A_6 will be $A_6 = A_2^* \oplus A_4 = h(HID_i \| S_{GW_j}).P \oplus S_{GW_j}.P$, After calculating it, the attacker can understand that it is a constant value related to fixed information obtained from the protocol steps. As a result, by obtaining A_6 from the protocol messages, i.e., A_9 and A_{10}, the user can be identified, because $A_6 = A_{10} \oplus A_9$ with a success probability of "1."

Therefore, we demonstrated that Sureshkumar *et al.*'s protocol does not resist traceability attack. In Section 5.3, we will show how the proposed protocol in this paper i.e. ECCbAS solves vulnerability against the traceability attack.

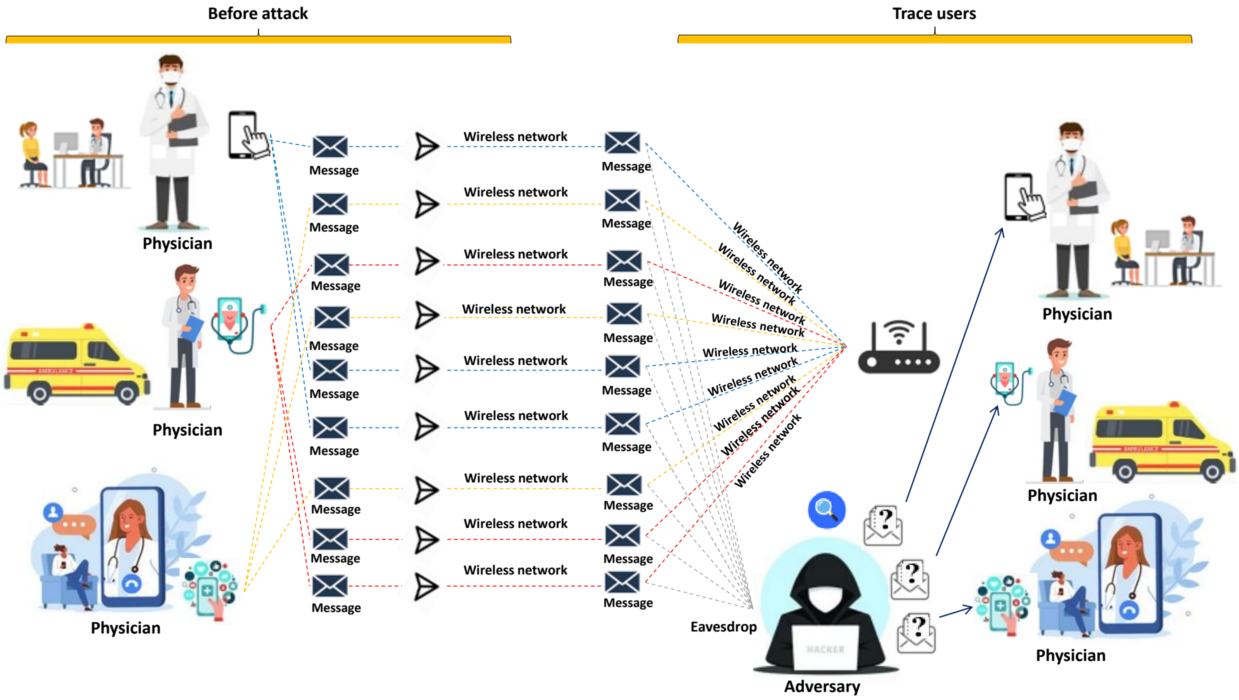


Figure 5: An example of traceability attack

4.2. Integrity Contradiction Attack

Any change in communication must be perceived by the recipient side of the message to be a violation of integrity. Figure 6 shows an example of integrity contradiction attack. The attacker can jeopardize the integrity of Sureshkumar *et al.*'s protocol by performing the following actions:

1. The message $M_1 = \langle A_7, A_9, A_{10}, T_1 \rangle$ is eavesdropped and attained by the adversary.
2. The adversary converts A_9 and A_{10} to $A_9^{/} = A_9 \oplus \Delta$ and $A_{10}^{/} = A_{10} \oplus \Delta$ respectively where Δ is an arbitrary value.
3. The adversary sends the values $A_9^{/}$ and $A_{10}^{/}$ to the gateway node instead of A_9 and A_{10}.
4. GW_j computes A_4^*, A_8^*, and A_2^{**} after receiving $A_9^{/}$ and $A_{10}^{/}$ from the U_i. It is clear that the GW_j is unable to detect any changes in A_2^{**} and A_8^* because the arbitrary value (Δ) will be removed with the exclusive-or operation after calculating $A_2^{**} = A_9^{/} \oplus \Delta \oplus A_{10}^{/} \oplus \Delta \oplus A_4$. As a result, the A_7^* relationship is established. Changes in A_8^* have a direct effect on A_{11}. Also, there is an A_{11} effect on the sk on the sensor side. Consequently, the key on the SN_k side will be $sk = r_s \cdot A'_{11}$ while A'_{11} is a diffrent value from original A_{11} value, resulting in a different secret key between the sensor, the gateway, and the user.

As a result, the gateway node and the sensor node are not aware of these changes. While all protocol parties should be able to evaluate the message integrity, any changes in the sent message must be noticed by the other party to the protocol. The reason for this problem is that the correctness of these messages is not checked at all on the recipient side of the message. The success probability of this attack is one, and its complexity is only one protocol run. In addition, we solve vulnerability against integrity contradiction attack with some changes in Sureshkumar *et al.*'s protocol that can be seen in Section 5.3.

4.3. De-synchronization attack

De-synchronization attack can be accomplished, for example, by executing actions that cause the shared secret values on the connection's parties to be timed to different values, resulting in a departure from concurrency together.

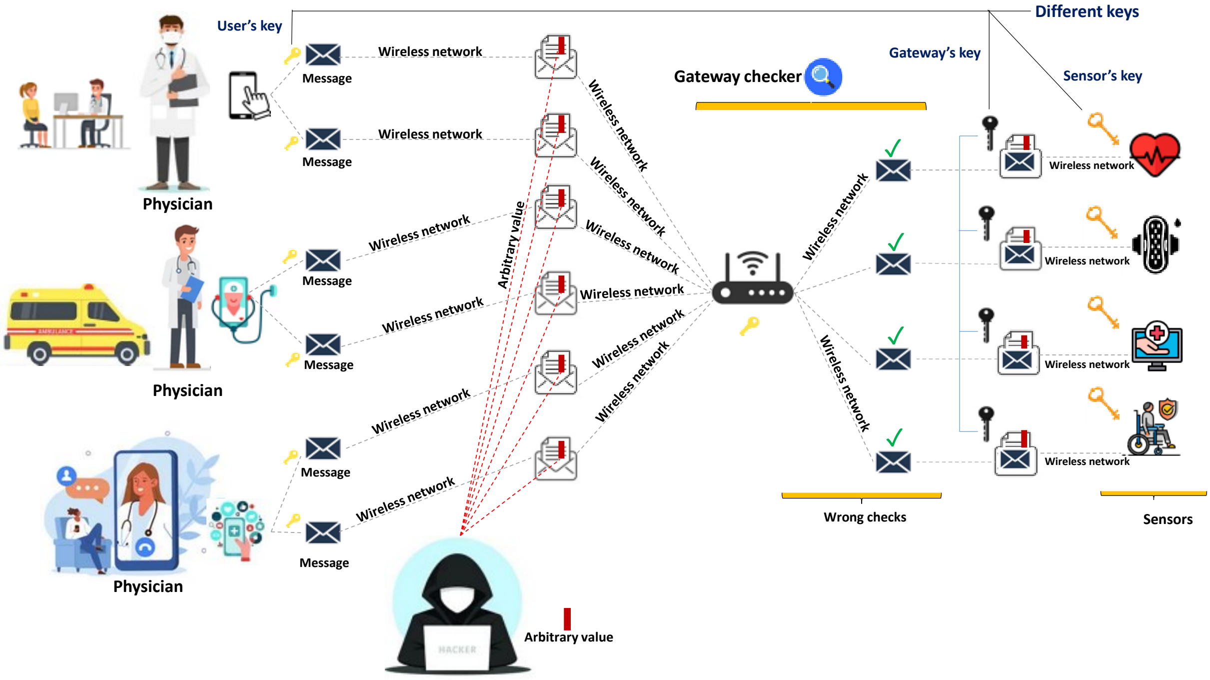


Figure 6: An example of Integrity Contradiction Attack

Figure 7 is an example of a de-synchronization attack. The adversary only needs to do the following to launch a de-synchronization attack against Sureshkumar *et al.*'s protocol:

1. The adversary eavesdrops $M_2 = \langle A_{12}, A_{11}, A_{15}, A_{16}, T_2 \rangle$ and A_{11} which is $A_{11} = r_g.A_8^*$.
2. Since, according to Sureshkumar *et al.*'s protocol steps, it can be seen that the integrity of A_{11} is not checked by the sensor node, so if instead of A_{11} in the message, the attacker changed A_{11} to $A'_{11} = A_{11} \oplus \Delta'$ where Δ' is an arbitrary value, then the secret key calculated in the sensor node will be as $sk = r_s.A'_{11} = r_s.(A_{11} \oplus \Delta')$. After that the sensor node sends $A_{19} = r_s.A_8$ to the GW_j.
3. Now if the adversary also changes A_{19} to $A'_{19} = A_{19} \oplus \Delta''$ where Δ'' is an arbitrary value, and sends A''_{19} to the gateway node, eventually the secret key in the gateway node will be $sk = r_g.A'_{19}$, so it can be seen that the secret key in gateway node is different from the secret key at the other side of the sensor node.
4. Similarly, if the adversary also changes A_{17} to $A'_{17} = A_{17} \oplus \Delta'''$, the key on the user side becomes $sk = r_u.A'_{17}$ in which case the key will differ on the three sides of the protocol.

Therefore, the Sureshkumar *et al.*'s protocol is not resistant to the above-explained de-synchronization attack. The success probability of this attack is one, and its complexity is only one protocol run. Furthermore, we solve vulnerability against this attack with some changes that can be seen in Section 5.4.

5. Proposed scheme: ECCbAS

In this section, we address the security flaws in the Sureshkumar *et al.*'s protocol, which led to the proposal of a new security protocol for the cloud-based healthcare system known as ECCbAS. Our goal in this section has been to improve the protocol's security issues rather than to create a completely new protocol from scratch. As a result, the primary structure of Sureshkumar *et al.*'s protocol has been retained in the proposed protocol. Similar to Sureshkumar *et al.*'s protocol, our proposed protocol (ECCbAS) includes a registration phase for sensor nodes (SN_K) and gateway

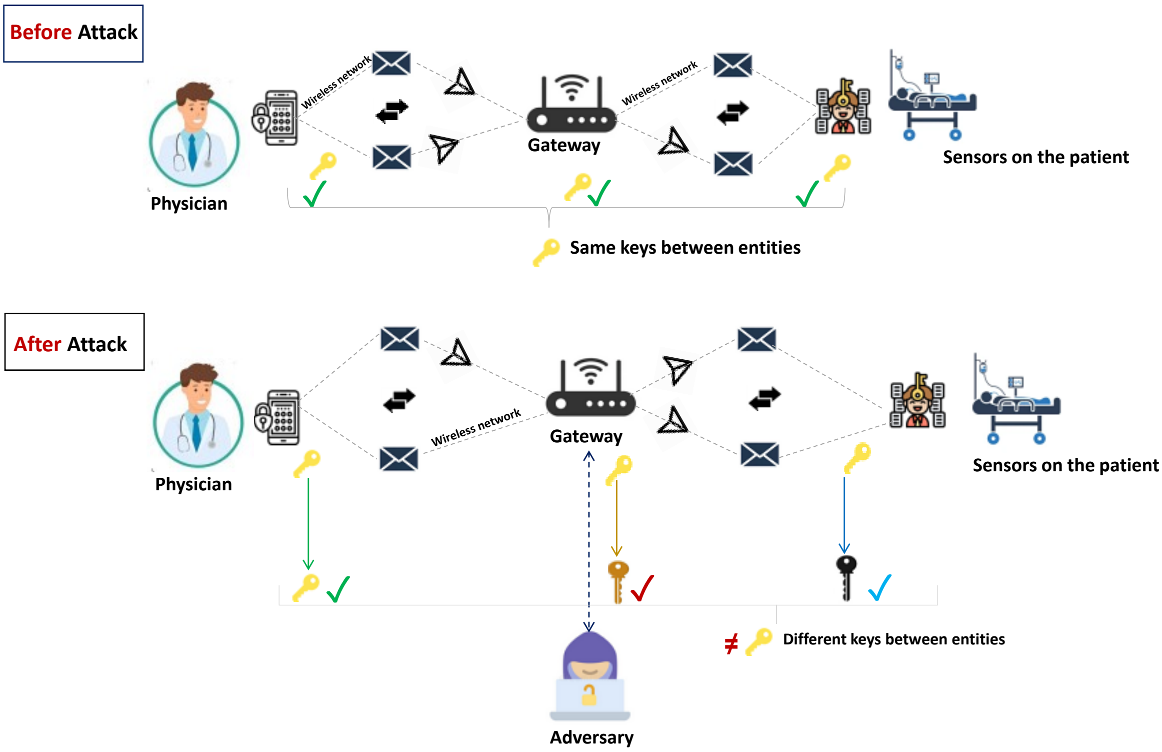


Figure 7: An example of a de-synchronization attack

Table 3
Common and secret values between entities in ECCbAS

User	Gateway node	sensor node
A_4	A_4	-
-	GW_{ID_j}	GW_{ID_j}
r_u	r_g	r_s
	S_{SN_K}	S_{SN_K}
SN_{ID_K}	SN_{ID_K}	SN_{ID_K}
-	A_{14}	A^*_{14}
-	A^*_{17}	A_{17}

nodes (GW_j) 2) User registration phase 3) Login and authentication phase and 4) phase of password change. The sensor addition phase of the ECCbAS is identical to that of Sureshkumar *et al.*'s protocol, which we omit to avoid repetition. Table 3 illustrates, some protocol values are kept secret and common among entities in ECCbAS.

5.1. Gateway and sensor node registration phase

The gateways (GW) and sensor nodes (SN) are entities that must be manually enrolled. The System Administrator (SA) accomplishes this phase by following the steps articulated below to enroll them.

1. SA chooses a GW_{ID_j} identity for GW_j, after that it computes and stores the value $S_{GW_j} = h(S_{SA} \| GW_{ID_j})$ in its database. Furthermore, SA stores $\langle GW_{ID_j}, S_{GW_j}, A_4 = S_{GW_j}.P \rangle$ in the memory of the gateway GW$_j$.

2. The SA determines an identity SN_{ID_K} for the K^{th} sensor node and calculates $S_{SN_K} = h(S_{SA} \| SN_{ID_k})$ then keeps $\langle SN_{ID_k}, S_{SN_k} \rangle$ and also stores this value in both sensor node and gateway node. The SA also stores

GW_{ID_j} in the memory of sensor node. The sensor node and gateway registration method are expressed in Algorithm 6.

Data: Information to be stored in entities memories.
Result: Registering sensors and gateway nodes

1: SA chooses identity GW_{ID_j} for GW_j.
2: SA computes $S_{GW_j} = h(S_{SA} \| GW_{ID_j})$ and saves it in its database in touch to GW_j.
3. Furthermore, SA stores $\langle GW_{ID_j}, S_{GW_j}, A_4 = S_{GW_j}.P \rangle$ in the memory of the gateway GW_j.
4: SA chooses an identity SN_{ID_K} for the K^{th} sensor node SN_K.
5: SA computes $S_{SN_k} = h(S_{SA} \| SN_{ID_K})$.
6: SA saves $\langle SN_{ID_k}, S_{SN_K} \rangle$ in the memory of both GW_j and SN_K.
7: The SA also stores GW_{ID_j} in the memory of sensor node.

Algorithm 6: ECCbAS's procedure for sensors and gateway nodes registration

5.2. User registration phase

A legitimate user is granted access to the detected data after successful user authentication. This event occurs when the gateway node reads the sensor's observed data, and the process of user registration is discussed further below:
1) The user U_i selects ID_i, PW_i, and computes its bio hashing $b_i = H(B_i)$ with its identity ID_i and password PW_i. After that U_i also computes $HID_i = h(ID_i \| b_i)$ and $HPW_i = h(PW_i \| b_i)$, and then sends $\langle HID_i, HPW_i, GW_{ID_j} \rangle$ to SA.
2) Once received the message, SA calculates A_1, A_2, A_3 and A_4 as $A_1 = h(HID_i \| HPW_i)$,$A_2 = h(HID_i \| S_{GW_j})$, $A_3 = A_2 \oplus A_1$, $A_4 = S_{GW_j}.P$, and then creates the SC=$\langle A_3, A_4, h(.), P \rangle$ and sends it to the user through a secure channel.
3) The user computes $HPID_i = h(HID_i \| PW_i)$, $A_5 = h(HID_i \| HPID_i)$, $A_2^* = A_3 \oplus A_1$, and $A_6 = A_2^* \oplus A_4$ after obtaining the smart card SC from SA.
The user smart card is SC=$\langle A_3$,A_5,A_6,$h(.)$, $P \rangle$ after replacing A_4 with A_6 in the smart card and storing A_5.

5.3. Login phase

As previously stated, a user wishing to obtain sensing data from a sensor SN_k must do the following via the gateway node. The details of this phase are explained below, and the procedure shows how the mentioned attacks can be solved with our improvements:

1. The user inputs his/her SC and enters ID_i, PW_i with the biometric B_i.
2. Then, $HID_i = h(ID_i \| b_i)$ and $HPID_i = h(HID_i \| PW_i)$ are calculated by the smart card. After that, the smart card computes $A_5^* = h(HID_i \| HPID_i)$ and checks whether $A_5^* \stackrel{?}{=} A_5$, if this equality does not hold, the session is terminated; otherwise, SC selects a random number $r_u \in F_q$ and calculates: $HPW_i = h(PW_i \| b_i)$, $A_1^* = h(HID_i \| HPW_i)$, $A_2 = A_3 \oplus A_1^* = h(HID_i \| S_{GW_j})$,$A_4 = A_6 \oplus A_2$, $A_8 = r_u.P$,$A_7 = (A_2 \| HID_i \| SN_{ID_k}) \oplus r_u.A_4$, and $A_9 = h(A_7 \| A_8 \| T_1)$, where T_1 represents the current timestamp.
3. Finally, the SC transfers the login message $M_1 = \langle A_7, A_8, A_9, T_1 \rangle$ to the GW_j node. In addition, Algorithms 7 shows the summary of user login procedure of ECCbAS.

Data: U_i Enters his or her own SC into the terminal, along with personal information such as $\langle ID_i, PW_i, B_i \rangle$
Result: User login

1. U_i inserts his/her own SC and enters its ID_i, PW_i, and B_i.
2. SC computes $b_i, HID_i, HPID_i, A_5^*$.
3. SC checks whether $A_5^* \stackrel{?}{=} A_5$
   If this equality is incorrect, SC will terminate the protocol; otherwise, SC selects r_u and calculates HPW_i,A_1^*, A_2,A_4, A_7, A_8, A_9.
4. At the end, SC transmits $M_1 = \langle A_7, A_8, A_9, T_1 \rangle$ to the GW_j.

Algorithm 7: The user login procedure in the ECCbAS

5.4. Authentication phase

The goal of this step is to authenticate the protocol entities as well as to generate a shared secret key between the user, the gateway, and the sensor node. The steps below demonstrate how to accomplish this:

1. $U_i \rightarrow GW_j$
   After receiving login message request, the gateway node GW_j calculates the time delay $\Delta T = T_2 - T_1$ by using the gateway node's timestamp T_2, and if the time delay ΔT is not sensible, the login and authentication procedure fails. If the time delay is acceptable, GW_j obtains $A_2^* \| HID_i^* \| SN_{ID_k}^*$ through computing $A_7 \oplus A_8.S_{GW_j}$. Then it determines whether $A_2^* \stackrel{?}{=} h(HID_i^* \| S_{GW_j})$. If so, it checks whether $A_9 \stackrel{?}{=} h(A_7 \| A_8 \| T_1)$. If it is ok, GW_j chooses a random number $r_g \in F_q$ and calculates: $A_{11} = r_g.A_8 = r_u.r_g.P$, $A_{12} = r_g.P$, $A_{13} = h(S_{SN_k}).A_{12}$, $A_{14} = h(GW_{ID_j} \| S_{SN_k}).P$, $A_{15} = h(A_{14} \| A_{12} \| A_{11} \| A_{16} \| T_2)$, and $A_{16} = A_8 \oplus A_{13}$. Then GW_j sends $M_2 = \langle A_{12}, A_{11}, A_{15}, A_{16}, T_2 \rangle$ to the SN_k.

2. $GW_j \rightarrow SN_k$
   When the sensor node SN_k receives M_2 message from the gateway node, confirms the time delay of $\Delta T = T_3 - T_2$ by its own timestamp, and if the time delay ΔT is incorrect, the protocol is terminated. Otherwise the sensor node SN_k calculates $A_{14}^* = h(GW_{ID_j} \| S_{SN_k}).P$, $A_{15}^* = h(A_{14}^* \| A_{12} \| A_{11} \| A_{16} \| T_2)$, and checks whether $A_{15}^* \stackrel{?}{=} A_{15}$. If this equality holds, the SN_k chooses a random number $r_s \in F_q$ and computes $A_{17} = r_s.A_{12}$, $A_{18} = h(A_{17} \| S_{SN_K} \| A_{19} \| A_{20} \| T_3)$, $A_{13}^* = h(S_{SN_k}).A_{12}$, $A_8^{**} = A_{16} \oplus A_{13}^*$, $A_{19} = r_s.A_8^{**}$, and $A_{20} = r_s.P$. Following this phase, the SN_k calculates the session key as $sk = r_s.A_{11}$ and sends the message $M_3 = \langle A_{19}, A_{18}, A_{20}, T_3 \rangle$ to the GW_j.

3. $SN_k \rightarrow GW_j$
   After receiving M_3 from the sensor node SN_k, the GW_j verifies delay $\Delta T = T_4 - T_3$ by its own timestamp T_4, and if this time delay ΔT is not correct, the gateway node rejects the procedure. Otherwise, it computes $A_{17}^* = r_g.A_{20}$ and checks whether $A_{18}^* = h(A_{17}^* \| S_{SN_k} \| A_{19} \| A_{20} \| T_3) \stackrel{?}{=} A_{18}$. If so, then the GW_j node computes $A_{21} = h(HID_i^* \| A_{17}^* \| A_8 \| A_4 \| T_4)$ and sends message M_4=$\langle A_{17}^*, A_{21}$,T$_4\rangle$ to the user. Furthermore, the gateway node computes the session key as $sk = r_g.A_{19}$.

4. $GW_j \rightarrow U_i$
   Following the receipt of message M_4 from the gateway GW_j node, the user computes $A_{21}^* = h(HID_i \| A_{17}^* \| A_8 \| A_4 \| T_4)$ and checks whether $A_{21}^* \stackrel{?}{=} A_{21}$. If so, the U_i will calculate the session key as $sk = r_u.A_{17}^*$.

The process of ECCbAS's login and authentication protocol is shown in Figure 8. Furthermore, Algorithms 8 shows the summary of the user authentication procedure of the ECCbAS.

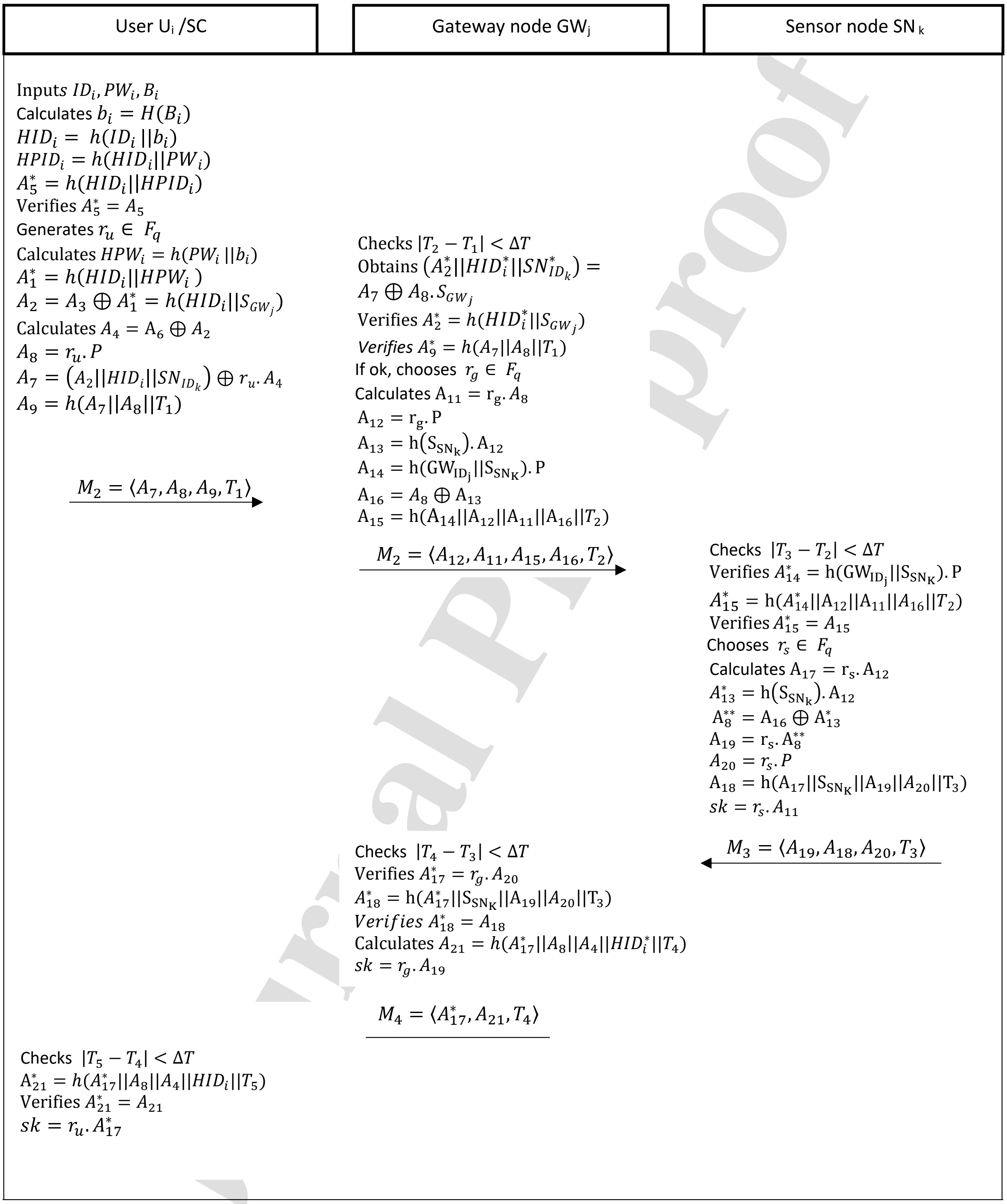


Figure 8: The login and authentication phase of ECCbAS over a public channel

Data: $M_1 = \langle A_7, A_8, A_9, T_1 \rangle$
Result: Authenticate the user

1. GW_j receives $M_1 = \langle A_7, A_8, A_9, T_1 \rangle$ from U_i.
2. After checking timestamp by GW_j, the GW_j obtains A_2^*,HID_i^*, $SN_{ID_k}^*$ as $A_7 \oplus A_8.S_{GW_j}$.
3. GW_j checks whether $A_2^* \stackrel{?}{=} h(HID_i^* \| S_{GW_j})$
   If this equality is not sensible, the GW_j will terminate the protocol; otherwise it checks whether $A_9 \stackrel{?}{=} h(A_7 \| A_8 \| T_1)$. If it is ok, GW_j selects r_g and calculatesA_{11}, A_{12}, A_{13}, A_{14}, A_{15} and A_{16}.
4. GW_j sends M_2 to SN_k.
5. After receiving M_2 from GW_j, SN_k checks timestamp and calculates A_{14}^* and A_{15}^*.
6. SN_K checks whether $A_{15}^* \stackrel{?}{=} A_{15}$
   If this equality was wrong, SN_k will terminate the protocol; otherwise, it chooses r_s and computes A_{17}, A_{18}, A_{13}^*, A_8^{**}, A_{19}, A_{20} and secret key (sk).
7. SN_k transmits M_3 to GW_j.
8. After getting the message M_3 from SN_k, GW_j checks timestamp and computes A_{17}^* and A_{18}^*.
9. GW_j checks whether $A_{18}^* \stackrel{?}{=} A_{18}$
   If this equality is not correct, SN_k will terminate the protocol; otherwise, it calculates A_{21}^* and secret key (sk).
10. GW_j sends M_4 to U_i.
11. After obtaining M_4 from GW_j, U_i checks timestamp and computes A_{21}^*.
12. U_i checks whether $A_{21}^* \stackrel{?}{=} A_{21}$
    If this equality was not true, U_i will terminate the protocol; otherwise, it computes secret key(sk).

Algorithm 8: The authentication procedure in the ECCbAS

5.5. Update Password

To increase security, passwords must be changed periodically. For this purpose, the following steps are considered in the ECCbAS:

1. U_i inputs his smart card into the terminal and enters his ID_i, PW_i with his biometric B_i.
2. A smart card (SC) calculates $b_i = H(B_i)$, $HID_i = h(ID_i \| b_i)$ and $HPID_= h(HID_i \| PW_i)$. SC also calculates $A_5^* = h(HID_i \| HPID_i)$ and checks whether $A_5^* \stackrel{?}{=} A_5$. When the equality is not true, SC terminates the session; otherwise, the SC allows U_i to enter a new password.
3. PW_i^{new} is the user's new password, which he enters into the SC.
4. The smart card calculates $HPID_i^{new} = h(HID_i \| PW_i^{new})$, $A_1^{new} = h(HID_i \| HPW_i^{new})$, $A_3^{new} = (A_3 \oplus A_1) \oplus A_1^{new}$, and $A_5^{new} = h(HID_i \| HPID_i^{new})$.

Finally, the SC replaces A_3 and A_5 with A_3^{new} and A_5^{new} respectively. Algorithm 9 shows how the password is updated in the ECCbAS.

Data: Personality information such as $\langle ID_i, PW_i, B_i \rangle$
Result: Update password

1. U_i inserts his smart card and enters ID_i, PW_i, and his biometric B_i.
2. SC calculates $b_i = H(B_i)$, $HID_i = h(ID_i \| b_i)$ and $HPID_i = h(HID_i \| PW_i)$.
3. SC also calculates $A_5^* = h(HID_i \| HPID_i)$
4. **if**($A_5^* == A_5$) **then**
   (a) U_i enters a new password using his SC.
   (b) U_i enters his new password PW_i^{new} into SC.
   (c) SC calculates $HPID_i^{new} = h(HID_i \| PW_i^{new})$, $A_1^{new} = h(HID_i \| HPW_i^{new})$, $A_3^{new} = (A_3 \oplus A_1) \oplus A_1^{new}$, $A_5^{new} = h(HID_i \| HPID_i^{new})$
   (d) SC substitutes old A_3 and A_5 with A_3^{new} and A_5^{new} respectively.
5. **else**
6. SC terminates the session.
7. **end if**

Algorithm 9: Algorithm of update password in the ECCbAS

6. ECCbAS Security Analysis

6.1. Informal security analysis

6.1.1. Anonymity

The identity of user ID_i is protected in all exchanged messages i.e. M_1, M_2, M_3 and M_4 of the proposed protocol using a cryptographic one-way hash function. Furthermore, during the registration phase, the user submits his ID_i to the system administrator after masking it with his/her biometric b_i using the hash function, so an insider can't obtain the user's identity. Thus the user's identity ID_i will not be revealed if an adversary tries to deduce this from the exchanged messages HID_i, A_3, and A_5. Because they are masked using hash functions with additional unknown secrets b_i, S_{GW_j}, and $HPID_i$. Moreover, checking the correctness of the guessed ID_i is extremely difficult. As a result, the attacker seems to have no means of confirming the user's ID_i or guessing it. Therefore, ECCbAS protects the anonymity of its users.

6.1.2. Off-line password guessing attack

All exchanged messages are protected using a one-way hash function with the user's password PW_i. Because the user submits his PW_i to SA after masking it with his/her biometric with the hash function during the registration phase, the insider cannot determine the user's password. Because the masking of PW_i requires additional secrets b_i, S_{GW_j}, and $HPID_i$ when using the hash function, it is extremely difficult to verify the consistency of the predicted PW_i. As a result, the attacker has no way of obtaining or guessing the password PW_i, making ECCbAS effective against off-line password guessing attacks.

6.1.3. Man In the Middle attack

When a protocol party is the target of a MITM attack, the adversary establishes separate connections with them and relays communications between them. Assume the attacker disrupts one of the initiator's messages, such as M_1. The attacker should then generate a new message with the timestamp T_1 and a random number. The message M_1 should have the $M_1 = \langle A_7, A_8, A_9, T_1 \rangle$ structure, where $A_7 = (A_2 \| HID_i \| SN_{ID_k}) \oplus r_u.A_4$ is protected by HID_i and A_2. A_2 also equals to $h(HID_i \| S_{GW_j})$. As a result, because the attacker is unable to recognize or guess the secret values i.e. HID_i or S_{GW_j}, s/he is unable to generate such $M_1 = \langle A_7, A_8 A_9, T_1 \rangle$ messages. As a result, the proposed scheme is secure and resistant to all types of man in the middle attacks.

6.1.4. Privileged insider attack

A privileged insider on the SA side can have access to the data about user U_i. The attacker is unable to guess the user identity ID_i despite having all of the registration details, including HID_i, HPW_i, and GW_{ID_j}. Because the user's

PW_i and biometric b_i, which cannot be deduced from HID_i, HPW_i, or GW_{ID_j}, are used to protect it.

6.1.5. Stolen smart card attack

If an attacker obtains all of the information in a user's smart cart SC after registration, the attacker knows A_3, A_4, A_5, A_6, and P, but the adversary does not retrieve user's identity ID_i. Since the user's identity ID_i is protected by the gateway's master secret S_{GW_j} and the attacker cannot calculate it from A_3, A_5 or A_6. Therefore, ECCbAS is resistant and secure against stolen smart cart attacks.

6.1.6. User impersonation attack

Given an adversary claims to be an honest user and attempts to impersonate another legal user, must construct a valid login message M_1 =(A_7, A_8, A_9, T_1). For this purpose and to calculate A_7, the adversary needs the correct values of A_2 which equals to $h(HID_i \| S_{GW_j})$ and HID_i. Since both of these values are unknown to the adversary, s/he cannot carry out the user impersonation attack. Therefore, ECCbAS also resists against user impersonation attacks.

6.1.7. Gateway node impersonation attack

The message M_2 is transferred from the GW_j to the SN_k that includes $A_{15} = h(A_{14} \| A_{12} \| A_{11} \| A_{16} \| T_2)$, in which $A_{14} = h(GW_{ID_j} \| S_{SN_k}).P$ consists of the two secret values GW_{ID_j} and S_{SN_k}. However, identifying both secret values at the same time makes the calculation impossible. Additionally the message $M_4 = \langle A^*_{17}, A_{21}, T_4 \rangle$ is passed on from the GW_i to the U_i that includes $A_{21} = h(HID^*_i \| A^*_{17} \| A_8 \| A_4 \| T_4)$, in which consists of a secret value such as HID^*_i which is unknown to the attacker. Therefore, ECCbAS resists against gateway node impersonation attacks.

6.1.8. Sensor node impersonation attack

Given that the adversary eavesdropped on the $M_2 = \langle A_{12}, A_{11}, A_{15}, A_{16}, T_2 \rangle$ from the GW_j and tries to impersonate SN_K. As a result, the adversary must compute the fabricated message $M_3 = \langle A^d_{19}, A^d_{18}, A^d_{20}, T_3 \rangle$ where: $A^d_{17} = r^d_s.A_{12}$, $A^*_{13} = h(S_{SN_k}).A_{12}$, $A^{**}_8 = A_{16} \oplus A_{13}$, $A^d_{19} = r^d_s.A^{**}_8$, and $A^d_{20} = r^d_s.P$, $A^d_{18} = h(A^d_{17} \| S_{SN_K} \| A^d_{19} \| A^d_{20} \| T_3)$. However, these calculations need the use of the secret value S_{SN_k}, which is unknown to the attacker. As a result, the attacker is unable to impersonate the SN_k.

6.1.9. Replay attack

Replay attacks are frequently used by attackers to impersonate a user's identity. Given that the malicious U_i obtains the old exchanged messages and sends them to the intended recipient unmodified during the current session, the protocol that uses timestamp can reject these types of messages sent by the adversary. As a result of the use of timestamps, ECCbAS is resistant to replay attacks.

6.1.10. Off-line sensor node identity guessing attack

Since the messages M_1, M_2, M_3 and M_4 do not include sensor identity in plain form, an attacker can't derive or guess SN_{ID_k} from public messages. Therefore, ECCbAS is secure against off-line sensor node guessing attacks.

6.1.11. Session key computation attack

Overall, the protocol's session key is used to encrypt messages sent across public channels between entities (our entities in this case are U_i, GW_j, and SN_k). The freshness of a session key is its most important feature, implying that it must be unique for each session. The entities user (U_i), gateway nodes (GW_j), and sensor nodes (SN_k) agree on the session key $sk = r_u.r_g.r_s.P$, which is based on the secret random numbers r_u, r_g, and r_s in the proposed protocol. As a result, the session key sk is generated using new values such as r_u, r_g, r_s. Furthermore, since the attacker lacks knowledge of the secret values r_u, r_g, and r_s, constructing the session key sk is impossible. As a result, ECCbAS is impervious to session key computation attacks.

6.1.12. Denial of service attack (DoS)

DoS attacks are theoretically possible at multiple network layers. However, because our protocol is based on challenge response, the attack is not possible. The user will always receive a rejection or confirmation message from the sensor node, ensuring that the obtained response message is genuine and not a DoS attack. Furthermore, because all messages have timestamps, a replay attack is also impossible. So, ECCbAS resists all kinds of DoS attacks.

6.1.13. Perfect Secrecy

Given the master secret key of sensor and gateway nodes, the attacker cannot compute all previously used session keys while still using known information, which is interpreted as forward secrecy in the proposed protocol. Because the session key is based on the secret random numbers r_u, r_g, and r_s generated by the three protocol sides in the session and the adversary does not have access to those values and also can not retrieve them from previously eavesdropped messages exchanged because they are all protected using ECC. Similarly, if the adversary knows the master secret key of sensor and gateway nodes, s/he cannot compute future session keys, resulting in backward secrecy. As a result, the protocol proposed in this paper achieves perfect secrecy.

6.1.14. Three-party mutual authentication

The proposed authentication protocol's final goal is to create a session key that allows each entity to communicate securely with others. The SN_k, GW_j, and U_i create a session key sk, which is then used for their mutual authentication in the proposed protocol.

6.2. Formal Security Analysis

6.2.1. Through Scyther

Scyther is a formal security protocol verification tool that was developed under the perfect cryptography assumption by Cremers (2008). This tool can determine a protocol's security requirements and vulnerabilities. The algorithms developed in the Scyther tool can provide benefits like:

1. Remarkable accomplishments that have enabled new models for protocol analysis, especially multi-protocol analysis.
2. A full feature is the powerful production of a finite description of an infinite number of model traces. The Security Protocol Description Language (SPDL) is used to specify the proposed scheme. The user, gateway, and sensor roles are described in this specification. Each role has its own sequence of circumstances such as receiving, sending, announcements, and claiming. Moreover, all claims defined in the Scyther tool are represented in Table 4.

We evaluate the security of the Sureshkumar *et al.*'s protocol through Scyther which shows that the Sureshkumar *et al.* 's protocol is not secure. The security verification results of the Sureshkumar *et al.*'s protocol are shown in Figure 9, which once again confirms the attacks presented in Section 4. The ECCbAS security confirmation results through Scyther are shown in Figure 10, which indicates that the ECCbAS fulfils all security requirements and that no attacks have been discovered. The SPDL implementation of ECCbAS is also shown in Figure 11.

6.2.2. Through ProVerif

The Proverif is a widely used tool for determining whether a cryptographic protocol fulfills the security principles or not. In this tool, many cryptographic functions are covered, including signatures, bit-commitment, symmetric and asymmetric encryption, and hash functions. Security objectives, such as communications affirmation and reachability attributes, can be assessed. For protocol analysis, infinite sessions and message space are modeled, and attack rebuilding is performed if some security characteristics are not fulfilled. In this section, we consider the security of ECCbAS through ProVerif. The ProVerif declarations of protocol to model ECCbAS and its security objectives, declarations of the necessary types, names, functions, events, and queries, and also sub-processes (macros), which are accomplished and processed using these declarations and sub-processes, are depicted in Figure 12 and can be carried out by ProVerif. The ProVerif security verification results of ECCbAS demonstrate that the security objectives of ECCbAS are met. Figure 13 shows the security verification results of ECCbAS through the ProVerif tool.

6.2.3. Through BAN logic

In 1990, Burrows, Abadi, and Needham developed a logic-based technique named BAN logic to verify the security of protocols in a formal approach. The protocols and their security objectives were specified in BAN logic, and it is derived whether the protocol participants believe the security goals or not. The notations which were used in the proof are represented in (BAN logic rules) where A and B represent the protocol participants and X and Y are some messages or concepts related to the protocol. The notations used for the BAN logic security proof of ECCbAS are seen in Table 5.

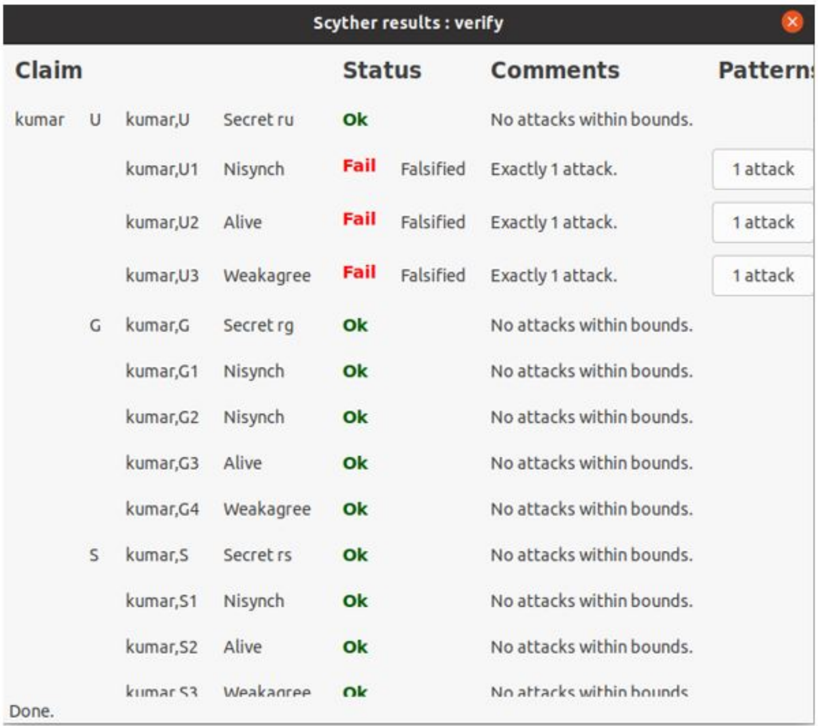


Figure 9: The security verification results of Sureshkumar *et al.*'s through Scyther tool

6.3. Used BAN logic rules

This section contains a list of some used BAN logic rules in this paper, which is depicted in Table 6.

6.4. Expressing ECCbAS protocol

In order to prove the security of ECCbAS through BAN logic, it must first be stated in the BAN logic form as below:

$M1 : U_i \rightarrow GW_j$: $A_7 = \{A_2, A_4, SN_{ID_K}, r_u\}_{HID_i}, A_8, A_9 = \{A_2, A_4, SN_{ID_K}, r_u, A_8, T_1\}_{HID_i}, T_1$

$M2 : GW_j \rightarrow SN_K$:$A_{11}, A_{12}, A_{15} = \{A_{11}, A_{16}, T_2\}_{S_{SNk}}, A_{16}, T_2$

$M3 : SN_K \rightarrow GW_j$: $A_{19}, A_{18} = \{A_{17}, A_{19}, A_{20}, T_3\}_{S_{SN_k}}, A_{20}, T_3$

$M4 : GW_j \rightarrow U_i$: $A^*_{17}, A_{21} = \{A^*_{17}, A_4, A_8, T_4\}_{HID_i}, T_4$

6.5. Idealizing ECCbAS protocol

In the second step, the ECCbAS messages should be idealized. In other words, messages that do not increase trust are deleted as below:

$IM1 : GW_j \triangleleft A_7 = \{A_2, A_4, SN_{ID_K}, r_u\}_{HID_i}, A_9 = \{A_2, A_4, SN_{ID_K}, r_u, A_8, T_1\}_{HID_i}$

$IM2 : SN_K \triangleleft A_{11}, A_{12}, A_{15} = \{A_{11}, A_{16}, T_2\}_{S_{SN_k}}, A_{16}$

$IM3 : GW_j \triangleleft A_{19}, A_{18} = \{A_{17}, A_{19}, A_{20}, T_3\}_{S_{SN_k}}, A_{20}$

$IM4 : U_i \triangleleft A^*_{17}, A_{21} = \{A_4, A^*_{17}, A_8, T_4\}_{HID_i}$

Table 4
Scyther tool security claims

Claims	Description
Secrecy	Secrecy states that no specific confidential information is exposed to the attacker, and even if this data is transferred over an insecure channel, various types of secrecy with distinctions can be provided.
Authentication	Authentication is the security aspect that has received the most attention in the field of security protocol analysis. However, despite the assurance of secrecy, there is no universal agreement on what authentication means. There is, in fact, a hierarchy of authentication traits, as Lowe (1997) shown Authentication is concerned with the notion that completing a protocol role assures that the network has at least one communication partner. In most circumstances, we wish to achieve a more concrete aim, such as ensuring that the other party is aware of our communication, that a protocol is in place, and that the messages are transferred as planned. The "Aliveness", "Synchronization", and "Agreement" characteristics of the Scyther tool refer to these hierarchies.
Aliveness	According to the definition, when an agent executes a role specification up to the claim event and believes he is communicating with a trusted agent, the intended communication partner has actually executed an event.
Synchronization	It requires a higher level of authentication. Synchronization requires the communication partner to send all incoming messages and the communication partner to receive the sent messages. This condition is consistent with the requirement that the actual message exchange take places exactly as specified in the protocol description. Synchronization criteria ensure that the protocol can behave according to predefined explanations even in the presence of the adversary.
Agreement	Agreement is another authentication criterion that focuses on the agreement reached between the parties in the protocol. The idea behind the Agreement criterion is that after the run of the protocol, the parties agree on the values of certain variables. Agreement is defined as a criterion that requires the content of the message to follow the message sent in accordance with the rules set by the protocol. As a result, after the protocol is executed, the content of the variables will be accurate as defined by the protocol. From this point of view, it is not possible to change the content of the message. If a message change occurs, the recipient of the message will notice the change.
Weakagree	This criterion results in a weak agreement in which the communication partners must ensure that they are communicating with each other in order to prevent one of them from being fabricated by the adversary.
Nisynch	Non-injective synchronization as defined in Lowe (1997) means that the receiving and sending events are executed by roles and in order and with the main content in question.
Niagree	Non-injective agreement on messages as defined in Lowe (1997) means that the sender and receiver agree on the secret values exchanged, and the results of the analysis justify the validity of this claim.
Empty	This claim is not verified, but simply ignored. This claim is only valid if Scyther is used as a back-end for other security verification tools.
Reachable	Once this claim is generated, the Scyther tool checks to see if this claim can be materialized at all. If there is a way in which this claim occurs, it is true. This claim can be useful for checking any obvious errors in the protocol specification and is actually inserted when the Scyther tool check mode is used.

6.6. ECCbAS security assumptions

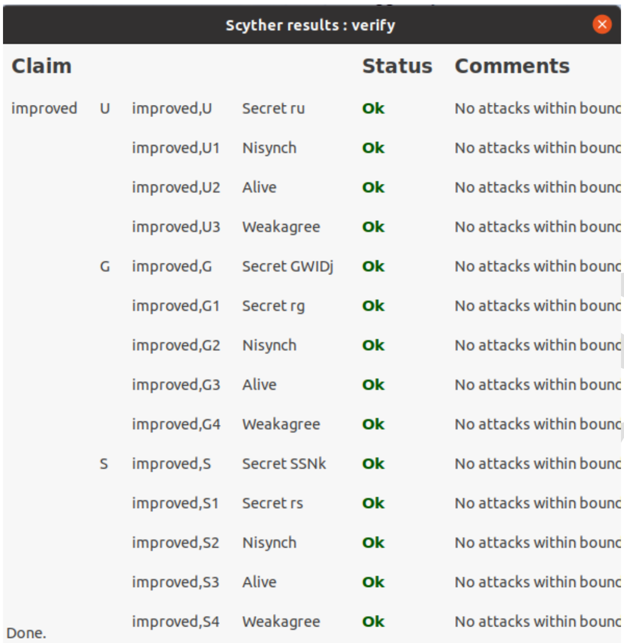
There are following assumptions in ECCbAS protocol.

$A1 : GW_j \mid\equiv GW_j \xleftrightarrow{A_4 = S_{GW_j}.P} U$

$A2 : GW_j \mid\equiv \#(r_g)$

$A3 : GW_j \mid\equiv U \mid\Rightarrow r_u$

$A4 : GW_j \mid\equiv SN_K \mid\Rightarrow r_s$


Scyther results : verify

Claim				Status	Comments
improved	U	improved,U	Secret ru	Ok	No attacks within bounc
		improved,U1	Nisynch	Ok	No attacks within bounc
		improved,U2	Alive	Ok	No attacks within bounc
		improved,U3	Weakagree	Ok	No attacks within bounc
	G	improved,G	Secret GWIDj	Ok	No attacks within bounc
		improved,G1	Secret rg	Ok	No attacks within bounc
		improved,G2	Nisynch	Ok	No attacks within bounc
		improved,G3	Alive	Ok	No attacks within bounc
		improved,G4	Weakagree	Ok	No attacks within bounc
	S	improved,S	Secret SSNk	Ok	No attacks within bounc
		improved,S1	Secret rs	Ok	No attacks within bounc
		improved,S2	Nisynch	Ok	No attacks within bounc
		improved,S3	Alive	Ok	No attacks within bounc
Done.		improved,S4	Weakagree	Ok	No attacks within bounc

Figure 10: The security verification results of ECCbAS through Scyther tool

Table 5
BAN logic notations used in this paper

Symbol	Explanation
$A \mid\equiv X$	means that A believes that X is true.
$A \triangleleft X$	means that if someone sends the message including the formula X, A will see it, possibly after performing some actions.
$A \mid\sim X$	means that the principle A sent the statement X.
$A \Rightarrow X$	means that A has control over formula X and if its value changes it is detectable for A.
$\#(X)$	means that X was recently generated. X has never been used before and may be a nonce.
$A \overset{K}{\leftrightarrow} B$	means that the two parties share the secret key K for their secure transmission, and that only the two are aware of K.
$A \overset{Y}{\rightleftharpoons} B$	means that the secret expression Y is shared by the two parties, A and B. Both may use it later to prove themselves to each other.

$A5: GW_j \mid\equiv \#(T_2)$

$A6: GW_j \mid\equiv \#(T_4)$

$A7: SN_K \mid\equiv GW_j \overset{S_{SN_k}}{\rightleftharpoons} SN_K$

$A8: SN_K \mid\equiv \#(T_3)$

$A9: SN_K \mid\equiv GW_j \mid\Rightarrow r_g$

$A10: SN_K \mid\equiv U_i \mid\Rightarrow r_u$

$A11: SN_K \mid\equiv \#(r_s)$

$A12: GW_j \mid\equiv GW_j \overset{S_{SN_K}}{\rightleftharpoons} SN_K$

$A13: U_i \mid\equiv GW_j \mid\Rightarrow (S_{GW_j})$

```
hashfunction H;
hashfunction ECC;
const xor : Function;
const con : Function;
const F512f : Function;
const M128m:  Function;
const E128e:  Function;
const P;
usertype Timestamp;
usertype Ticket;
protocol @oracle (X)
{
role X {
   var Y:Agent;
   const P;
   recv_!X1(X, X, ECC(X,ECC(Y,P)));
   send_!X2( X,X , ECC(Y,ECC(X,P)) );
     }
}
protocol improved(U,G,S){
role U{
     fresh ru: Nonce;
     var rg: Nonce;
     var rs: Nonce;
     fresh T1: Timestamp;
     var T4 : Timestamp;
     secret  HIDi;
     secret HPWi;
     secret IDi;
     secret PWi;
     secret Bi;
     secret SNidk;
     secret A3;
     secret A4;
     secret A6;
     macro bi=H(Bi);
     macro HIDi=H(con(IDi,bi));
     macro HPID=H(con(HIDi,PWi));
     macro HPWi=H(con(PWi,bi));
     macro A1star=H(con(HIDi,HPWi));
     macro A2=xor(A3,A1star);
     macro A4=xor(A6,A2);
     macro A8=ECC(ru,P);
macro A21star=H(con(con(con(con(A8,A4),A17star),HIDi),T4));
var A17star,A21: Ticket;
send_1 (U,G, A7,A8,A9,T1);
recv_4 (G,U,A21,A17star,T4);
match(A21star,A21);
claim_U (U, Secret, ru);
claim_U (U, Nisynch );
claim_U (U, Alive );
claim_U (U, Weakagree);
}
role G{
fresh rg: Nonce;
fresh T2: Timestamp;
var T3: Timestamp;
var ru: Nonce;
var rs: Nonce;
var T1: Timestamp;
fresh T4: Timestamp;
secret SNidk;
secret IDi;
secret PWi;
secret SGWj;
secret GWIDj;
secret SSNk;
secret A4;
secret A3;
secret A6;
secret Bi;
macro A2star=F512f(ECC(xor(A7,A8),SGWj));
macro HIDstar=M128m(ECC(xor(A7,A8),SGWj));
macro SNidkstar=E128e(ECC(xor(A7,A8),SGWj));
macro A2dablstar=xor(HIDstar,SGWj);
macro A9star=H(con(con(T1,A7),A8));
macro A11=ECC(rg,A8);
macro A12=ECC(rg,P);
macro A13=ECC(H(SSNk),A12);
macro A14=ECC(H(con(GWIDj,SSNk)),P);
macro A16=xor(A8,A13);
macro A15=H(con(con(con(con(A14,T2),A12),A11),A16));
macro A17star=ECC(rg,A20);
macro A18star=H(con(con(con(con(A17star,SSNk),A19),A20),T3));
macro A21=H(con(con(con(con(A8,A4),A17star),HIDstar),T4));
var A19,A18,A20,A9;
recv_1 (U,G, A7,A8,A9,T1);
match(A9star,A9);
send_2 (G,S,A12,A11,A15,A16,T2);
recv_3 (S,G, A19,A18,A20,T3 );
match(A18,A18star);
send_4 (G,U,A21,A17star,T4);
claim_G (G, Secret, GWIDj);
claim_G (G, Secret, rg);
claim_G (G, Nisynch );
claim_G (G, Alive );
claim_G(G, Weakagree);
}
role S{
fresh T3: Timestamp;
var T2: Timestamp;
fresh rs: Nonce;
var rg: Nonce;
var ru: Nonce;
secret GWIDj;
secret SSNk;
secret SGWj;
secret SNidk;
secret A4;
macro A14star=ECC(H(con(GWIDj,SSNk)),P);
macro
A15star=H(con(con(con(con(A14star,T2),A12),A11),A16));
macro A17=ECC(rs,A12);
macro A13star=ECC(H(SSNk),A12);
macro A8dabelstar=xor(A16,A13star);
macro A19=ECC(rs,A8dabelstar);
macro A20=ECC(rs,P);
macro A18=H(con(con(con(con(A17,SSNk),T3),A19),A20));
var A15;
recv_2 (G,S,A12,A11,A15,A16,T2);
match(A15,A15star);
send_3 (S,G, A19,A18,A20,T3);
claim_S (S, Secret,SSNk);
claim_S (S, Secret,rs);
claim_S (S, Nisynch );
claim_S (S, Alive );
claim_S(S, Weakagree);
}
}
```

Figure 11: SPDL implementation of ECCbAS

$$A14 : U_i \mid\equiv U_i \overset{HID_i}{\rightleftharpoons} GW_j$$

$$A15 : GW_j \mid\equiv GW_j \overset{HID_i}{\rightleftharpoons} U_i$$

$$A16 : U_i \mid\equiv \#(T_4)$$

$$A17 : GW_j \mid\equiv \#(T_3)$$

$$A18 : SN_k \mid\equiv \#(T_2)$$

$$A19 : U_i \mid\equiv GW_j \mid\Rightarrow r_g$$

$$A20 : U_i \mid\equiv SN_K \mid\Rightarrow r_s$$

$$A21 : U_i \mid\equiv \#(r_u)$$

6.7. ECCbAS security goals

If it can be demonstrated using BAN logic that ECCbAS achieves the following goals, it demonstrates that ECCbAS is secure.

```
(*-channels-*)
free ch: channel.
free sch0: channel [private].
type Nonce.
type TimeStamp.
(*-session keys-*)
free Sku: bitstring [private].
free Sks: bitstring [private].
free Skg: bitstring [private].
(*-constant-*)
free IDi: bitstring [private].
free PWi: bitstring [private].
free Bi: bitstring [private].
free SGWj: bitstring [private].
free SSNk: bitstring [private].
free SNidk: bitstring [private].
free GWIDj: bitstring [private].
free A3: bitstring [private].
free A4: bitstring [private].
free A6: bitstring [private].
free P:bitstring.
(* Type Converter *)
fun nonce2bitstring(Nonce): bitstring [data,typeConverter].
(*-functions-*)
fun H(bitstring): bitstring.
fun xor(bitstring , bitstring): bitstring.
fun con(bitstring,bitstring): bitstring.
fun ECC(bitstring,bitstring): bitstring.
fun F512f(bitstring): bitstring.
fun M128m(bitstring): bitstring.
fun E128e(bitstring): bitstring.
(*-equations-*)
equation forall m: bitstring, n:bitstring;
xor(xor(m,n),n)=m.xor(xor(m,n),n)=m.
event beginUserA.
event endUserA.
event beginGatewayNodeA.
event endGatewayNodeA.
event beginSensorNodeA.
event endSensorNodeA.
(*-queries-*)
weaksecret IDi.
query attacker(new uru).
query attacker(new grg).
query attacker(new srs).
query inj-event(endUserA)==>inj-event(beginUserA).
query inj-event(endGatewayNodeA)==>inj-event(beginGatewayNodeA)
query inj-event(endSensorNodeA)==>inj-event(beginSensorNodeA).
(****************** User rule ******************)
let UserNode=
new T1:bitstring;
new uru:Nonce;
let ut = nonce2bitstring(uru) in
let bi=H(Bi) in
let HIDi=H(con(IDi,bi)) in
let HPWi=H(con(PWi,bi)) in
let A1star=H(con(HIDi,HPWi)) in
let A2star=xor(A3,A1star) in
let A4star=xor(A6,A2star) in
let A8=ECC(ut,P) in
let A7=ECC(xor(con(con(A2star,HIDi),SNidk),ut),A4star) in
let A9=H(con(con(T1,A7),A8)) in
event beginUserA;
out(ch,(A7,A8,A9,T1));
in (ch,(A17star:bitstring,A21:bitstring,T4:bitstring));
if A21star=H(con(con(con(A8,A4star),A17star),HIDi),T4) then
let Sku=ECC(ut,A17star) in
event endUserA.
(****************** Gateway Node rule ******************)
let GetwayNode=
in(ch,(A7:bitstring,A9:bitstring,T1:bitstring));
let A2star=F512f(ECC(xor(A7,A8),SGWj)) in
let HIDstar=M128m(ECC(xor(A7,A8),SGWj)) in
let SNidkstar=E128e(ECC(xor(A7,A8),SGWj)) in
let A2dablstar=xor(HIDstar,SGWj) in
let A9star=H(con(con(T1,A7),A8)) in
new grg:Nonce;
let gt = nonce2bitstring(grg) in
new T2:bitstring;
let A11= ECC(gtlet UserNode=
new T1:bitstring;
new uru:Nonce;
let ut = nonce2bitstring(uru) in
let bi=H(Bi) in
let HIDi=H(con(IDi,bi)) in
let HPWi=H(con(PWi,bi)) in
let A1star=H(con(HIDi,HPWi)) in
let A2star=xor(A3,A1star) in
let A4star=xor(A6,A2star) in
let A8=ECC(ut,P) in
let A7=ECC(xor(con(con(A2star,HIDi),SNidk),ut),A4star) in
let A9=H(con(con(T1,A7),A8)) in
event beginUserA;
out(ch,(A7,A8,A9,T1));
in (ch,(A17star:bitstring,A21:bitstring,T4:bitstring));
let A21star=H(con(con(con(con(A8,A4star),A17star),HIDi),T4)) in
if A21star=A21 then
let Sku=ECC(ut,A17star) in
event endUserA.
(****************** Gateway Node rule ******************)
let GetwayNode=
in(ch,(A7:bitstring,A8:bitstring,A9:bitstring,T1:bitstring));
let A2star=F512f(ECC(xor(A7,A8),SGWj)) in
let HIDstar=M128m(ECC(xor(A7,A8),SGWj)) in
let SNidkstar=E128e(ECC(xor(A7,A8),SGWj)) in
let A2dablstar=xor(HIDstar,SGWj) in
let A9star=H(con(con(T1,A7),A8)) in
new grg:Nonce;
let gt = nonce2bitstring(grg) in
new T2:bitstring;
let A11= ECC(gt,A8) in
let A12= ECC(gt,P) in
let A13= ECC(H(SSNk),A12) in
let A14= ECC(H(con(GWIDj,SSNk)),P) in
let A16=xor(A8,A13) in
let A15=H(con(con(con(con(A14,T2),A12),A11),A16)) in
event beginGatewayNodeA;
out (ch,(A12,A11,A15,A16,T2));
in (ch,(A19:bitstring,A18:bitstring,A20:bitstring,T3:bitstring));
let A17star=ECC(gt,A20) in
if  A18=H(con(con(con(con(A17star,SSNk),A19),A20),T3)) then
new T4:bitstring;
let A21=H(con(con(con(con(A8,A4),A17star),HIDstar),T4)) in
let Skg=ECC(gt,A19) in
out(ch,(A17star,A21,T4));
event endGatewayNodeA.
(****************** Sensor Node rule ******************)
let SensorNode=
in (ch,(A12:bitstring,A11:bitstring,A15:bitstring,A16:bitstring,T2:bitstring));
let A14star=ECC(H(con(GWIDj,SSNk)),P) in
let A15star=H(con(con(con(con(A14star,T2),A12),A11),A16)) in
if A15= H(con(con(con(con(A14star,T2),A12),A11),A16)) then
new srs:Nonce;
let st = nonce2bitstring(srs) in
let A17=ECC(st,A12) in
let A13star=ECC(H(SSNk),A12) in
let A8dabelstar=xor(A16,A13star) in
let A19=ECC(st,A8dabelstar) in
let A20=ECC(st,P) in
new T3:bitstring;
let A18=H(con(con(con(con(A17,SSNk),T3),A19),A20)) in
let Sks=ECC(st,A11) in
event beginSensorNodeA;
out(ch,(A19,A18,A20,T3));
event endSensorNodeA.
(****************** process ******************)
process((!UserNode) | (!GetwayNode) | (!SensorNode))
```

Figure 12: ProVerif code implementation of ECCbAS

$G1\ :\ U_i\ |\equiv sk$

$G2\ :\ GW_j\ |\equiv sk$

$G3\ :\ SN_K\ |\equiv sk$

6.8. Deduction of ECCbAS security goals

Using $IM4$, $A14$ and based on $Rule_1$, we have $D1\ :\ Ui\ |\equiv GW_j\ |\sim \{A_4, A^*_{17}, A_8, T_4\}$. Given $A16$, based on $Rule_5$, we get $D2\ :\ Ui\ |\equiv \#(A_4, A^*_{17}, A_8, T_4)$. Using $D1$, $D2$ based on $Rule_3$, we deduced that $D3\ :\ Ui\ |\equiv GW_j\ |\equiv \{A_4, A^*_{17}, A_8, T_4\}$. Given $D3$, based on $Rule_9$, we get $D4\ :\ Ui\ |\equiv GW_j\ |\equiv A^*_{17}$. Since $A^*_{17} = r_g.P$, using $A19$, we can get $D5\ :\ U_i\ |\equiv GW_j| \Rightarrow A^*_{17}$. Given $D5$ and $D3$ based on $Rule_4$, we have $D6\ :\ U_i\ |\equiv A^*_{17}$. Using $A21$ and $D6$, we can get $D7\ :\ U_i\ |\equiv r_u.A^*_{17} = sk$ which is the $G1$ security goal.

Using $IM3$, $A12$ and based on $Rule_1$, we have $D7\ :\ GW_j\ |\equiv SN_K\ |\sim \{A_{17}, A_{19}, A_{20}, T_3\}$. Given $A17$, based on $Rule_5$, we get $D8\ :\ GW_j\ |\equiv \#(A_{17}, A_{19}, A_{20}, T_3)$. Using $D7$, $D8$ based on $Rule_3$, we deduced that $D9\ :\ GW_j\ |\equiv SN_K\ |\equiv \{A_{17}, A_{19}, A_{20}, T_3\}$. Given $D9$, based on $Rule_9$, we get $D10\ :\ GW_j\ |\equiv SN_K\ |\equiv A_{19}$. Since $A_{19} = r_s.A_8$, using $A4$, we can get $D11\ :\ GW_j\ |\equiv SN_K| \Rightarrow A_{19}$. Given $D10$ and $D11$ based on $Rule_4$, we have $D12\ :\ GW_j\ |\equiv A_{19}$. Using $A2$ and $D12$, we can get $D13\ :\ GW_j\ |\equiv r_g.A_{19} = sk$ which is the $G2$ security goal.

Using $IM2$, $A7$ and based on $Rule_1$, we have $D14\ :\ SN_K\ |\equiv GW_j\ |\sim \{A_{11}, A_{16}, T_2\}$. Given $A18$, based on $Rule_5$, we get $D15\ :\ SN_K\ |\equiv \#(\{\}A_{11}, A_{16}, T_2\}$. Using $D14$ and $D15$ based on $Rule_3$, we deduced that $D16\ :\ SN_K\ |\equiv$

Verification summary:

- Weak secret ID_i is true.
- Query not attacker(uru[!1 = v]) is true.
- Query not attacker(grg[T1_1 = v,A9_1 = v_1,A8_1 = v_2,A7_1 = v_3,!1 = v_4]) is true.
- Query not attacker(srs[T2_1 = v,A16_1 = v_1,A15_1 = v_2,A11_1 = v_3,A12_1 = v_4,!1 = v_5]) is true.
- Query inj-event(endUserA) ==> inj-event(beginUserA) is true.
- Query inj-event(endGatewayNodeA) ==> inj-event(beginGatewayNodeA) is true.
- Query inj-event(endSensorNodeA) ==> inj-event(beginSensorNodeA) is true.

Figure 13: The security verification results of ECCbAS through ProVerif tool

Table 6
Used BAN logic rules

Rule	Explanation
$Rule_1 : \frac{A \mid\equiv (A \overset{K}{\leftrightarrow} B), A \lhd \{X\}_K}{A \mid\equiv B \mid\sim X}$	A believes that K is shared by B and sees X which is encrypted with K, then it is deduced that A believes that B has sent X.
$Rule_2 : \frac{A\ selects\ random\ X}{A \mid\equiv \#(X)}$	If A selects a random number, it is deduced that A believes that X is fresh.
$Rule_3 : \frac{A \mid\equiv \#(X), A \mid\equiv B \mid\sim X}{A \mid\equiv B \mid\equiv X}$	If A believes the X is fresh and A believes B has sent X, then it is deduced that A believes that B believes X.
$Rule_4 : : \frac{A \mid\equiv B \Rightarrow X, A \mid\equiv B \mid\equiv X}{A \mid\equiv X}$	If A believes that B has control over the X and also believes that B believes X, then it is deduced that A believes X.
$Rule_5 : \frac{A \mid\equiv \#(X)}{A \mid\equiv \#(X, Y)}$	If A believes that one part of an expression i.e. X is recent, then it is deduced that A believes the entire expression i.e. (X, Y) is also recent.
$Rule_6 : \frac{A \mid\equiv \#(X), A \mid\equiv B \mid\equiv X}{A \mid\equiv A \overset{K}{\rightleftharpoons} B}$	If A believes that the formula X is fresh, and A believes that B believes X, which is an important element of the session key, then it is deduced that A believes that they share the session key K with B.
$Rule_7 : \frac{A \mid\equiv X, A \mid\equiv Y}{A \mid\equiv (X, Y)}$	If A believes to formulas X and Y, then it is deduced A believes any combination of X and Y.
$Rule_8 : \frac{A \mid\equiv B \mid\sim (X, Y)}{A \mid\equiv B \mid\sim X}$	If A believes that B has sent (X, Y), then it is deduced A believes B has sent any part of it i.e. X and Y.
$Rule_9 : \frac{A \mid\equiv B \mid\equiv (X, Y)}{A \mid\equiv B \mid\equiv X}$	If A believes that B believes (X, Y), then it is deduced A believes B has believed any part of it i.e. X or Y.
$Rule_{10} : \frac{A \mid\equiv (X, Y)}{A \mid\equiv X}$	If A believes (X, Y), then it is deduced A believes any part of it i.e. X or Y.
$Rule_{11} : \frac{A \lhd (X, Y)}{A \lhd X}$	If A received (X, Y), then it is deduced A has received any part of it i.e. X or Y.

$GW_j \mid\equiv \{A_{11}, A_{16}, T_2\}$. Given $D16$, based on $Rule_9$, we get $D17 : SN_K \mid\equiv GW_j \mid\equiv A_{11}$. Using $A11$ and $D17$, we can get $D18 : SN_K \mid\equiv r_s.A_{11} = sk$ which is the $G3$ security goal.

7. Comparison and evaluation

7.1. Security comparison

Table 7 compares the proposed protocol with other protocols i.e. Wu et al. (2017); Li et al. (2016); He et al. (2015); Sureshkumar et al. (2019); Khan and Kumari (2014) in depth. Replay attack, privilege insider attack, untraceability

Table 7
Security comparison of ECCbAS with recent similar protocols

Protocols	A_2	A_2	A_3	A_4	A_5	A_6	A_7
Wu et al. (2017)	✗	✓	✓	✓	✗	✓	✓
Khan and Kumari (2014)	✓	✓	✓	✓	✗	✓	✓
Li et al. (2016)	✓	✗	✓	✓	✓	✓	✓
He et al. (2015)	✓	✗	✓	✓	✗	✗	✓
Sureshkumar et al. (2019)	✓	✓	✗	✗	✓	✓	✗
ECCbAS	✓	✓	✓	✓	✓	✓	✓

A_1: Replay attack resistance; A_2: Privilege insider attack resistance;
A_3: Overcomes the session key attack; A_4: Untraceability;
A_5: Impersonation attack resistance; A_6: Forward secrecy; A_7:De-synchronization attack resistance;
✓: Resistant ✗ : Vulnerable

Table 8
Computational cost comparison of ECCbAS with recent similar protocols (in milliseconds)

Protocols	Total Computational Cost of GW, SN, U
Wu et al. (2017)	$20T_h + 8T_{e/d} = 79.6$ ms
Khan and Kumari (2014)	$20T_h + 2T_{e/d} = 27.4$ ms
Li et al. (2016)	$18T_h + 10T_{e/d} = 96$ ms
He et al. (2015)	$7T_h + 9T_{e/d} = 81.8$ ms
Sureshkumar et al. (2019)	$18T_h + 17T_p = 16.514$ ms
ECCbAS	$18T_h + 15T_p = 15.63$ ms

and impersonatation attacks are such vulnerabilities in these protocols. Table 7 shows that ECCbAS resists to a wide range of security threats and has several functional requirements that make it more robust. (✓) in Table 7 indicates that the scheme resists against an attack, whereas (✗) indicates that the scheme does not resist an attack.

7.2. Computational cost comparison

Authentication schemes in MWSNs are designed to be lightweight in terms of computational cost due to energy constraints Wu et al. (2017). The proposed scheme uses a hash function and an elliptic curve cryptosystem, both of which are lightweight when compared to other operations like public key cryptographic functions and symmetric key encryption/decryption. Based on Sureshkumar et al. (2019), we used the hash function and symmetric key encryption/decryption running times of T_h=0.5 ms and $T_{e/d} = 8.7$ ms, respectively. $T_p = 0.442$ ms is the running time for elliptic curve point multiplication. Table 8 compares the computational costs of ECCbAS and existing protocols for gateway node (GW_j), sensor node (SN_k), and user (U_i). Sensor nodes have a much lower computational capacity than gateway nodes. The computational cost of the SN nodes should be whittled down to improve efficiency. Our protocol is more effective than other methods, as shown in Table 8, and also Figure 14 which demonstrates the comparison of the total computational cost of ECCbAS with other similar protocols. It should be noted that computational and communication costs are calculated only for the login and authentication phases of protocols.

7.3. Communication costs comparison

The total number of bits needed to transmit messages during the login authentication process is referred to as communication cost. Table 9 compares the communication cost of ECCbAS to that of the similar protocols recently proposed. We assume that the password, identity are 128 bits, output hash value of the hash function (SHA-256) and random numbers are all 256 bits and timestamps are all 32 bits. ECC (Elliptic Curve Cryptography) point and symmetric encryption (AES) have output lengths of 256 bits and 128 bits, respectively. In ECCbAS, U_i sends one elliptic curve cryptography point, one hash value, one timestamp and one 512 bit mixed message ($256 + 256 + 32 + 512 = 1056$ bits) to the GW_j and the GW_j sends two elliptic curve cryptography points with one hash value and one 256 bit mixed message and one timestamp ($512 + 256 + 256 + 32 = 1056$ bits) towards the SN_k and SN_k sends two elliptic curve cryptography point with one hash value and one timestamp ($512+256+32 = 800$ bits) to the GW_j, and the GW_j sends one elliptic curve cryptography point with one hash value and one timestamp ($256 + 256 + 32 = 544$ bits) to the U_i.

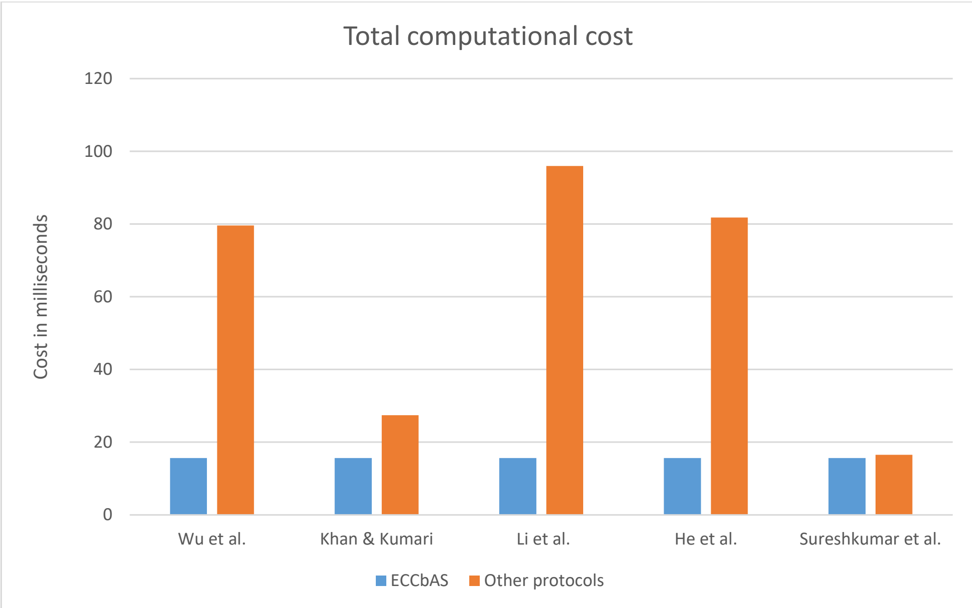


Figure 14: Comparison of total computational cost of ECCbAS with recent similar protocols

Table 9
Communication cost comparison of ECCbAS with recent similar protocols

Protocols	Total Communication Cost of GW_j, SN_K, U_i (in bits)
Wu et al. (2017)	$640 + 1280 + 512 = 2432$
Khan and Kumari (2014)	$1088 + 576 + 416 = 2080$
Li et al. (2016)	$320 + 160 + 288 = 768$
He et al. (2015)	$160 + 160 + 288 = 608$
Sureshkumar et al. (2019)	$1536 + 800 + 800 = 3168$
ECCbAS	$1600 + 800 + 1056 = 3456$

Consequently, the total communication cost of ECCbAS is equal to 3456 bits. Table 9 compares the communication cost of ECCbAS with ones of similar recent protocols. Even though the protocols in Khan and Kumari (2014); Li et al. (2016); He et al. (2015); Sureshkumar et al. (2019); Wu et al. (2017) have lower communication costs than ECCbAS, they did not address security issues like identity fraud, privileged insider attacks, or all kinds of impersonation attacks.

7.4. Storage cost comparison

Sensor nodes typically have lower storage capacities than gateway nodes. The sensor node's storage capacity must be reduced as much as possible. The ciphertext has a length of 128 bits. The hash function output and random numbers are 256 bits, password, and identity are of 128 bit length. Table 10 and Figure 15 show the storage costs for ECCbAS compared to Sureshkumar *et al.* and other similar protocols.

8. Conclusion

In this paper, we showed de-synchronization, traceability, and integrity contradiction attacks against Sureshkumar *et al.*'s protocol. All the attacks described in this paper have a success probability of "1", and their complexity is only one run of the protocol. Moreover, we evaluated the Sureshkumar *et al.*'s protocol security using the Scyther tool, the results of which also show the lack of complete security of this scheme. In addition, we proposed ECCbAS, a novel secure cloud-based RFID authentication protocol for use in healthcare systems. The protocol's informal and formal security analysis revealed that it provides adequate protection against a variety of attacks, especially the attacks

Table 10
Storage cost comparison of ECCbAS with recent similar protocols

Protocols	Storage Cost(in bits)
Wu et al. (2017)	256
Khan and Kumari (2014)	384
Li et al. (2016)	256
He et al. (2015)	256
Sureshkumar et al. (2019)	512
ECCbAS	512

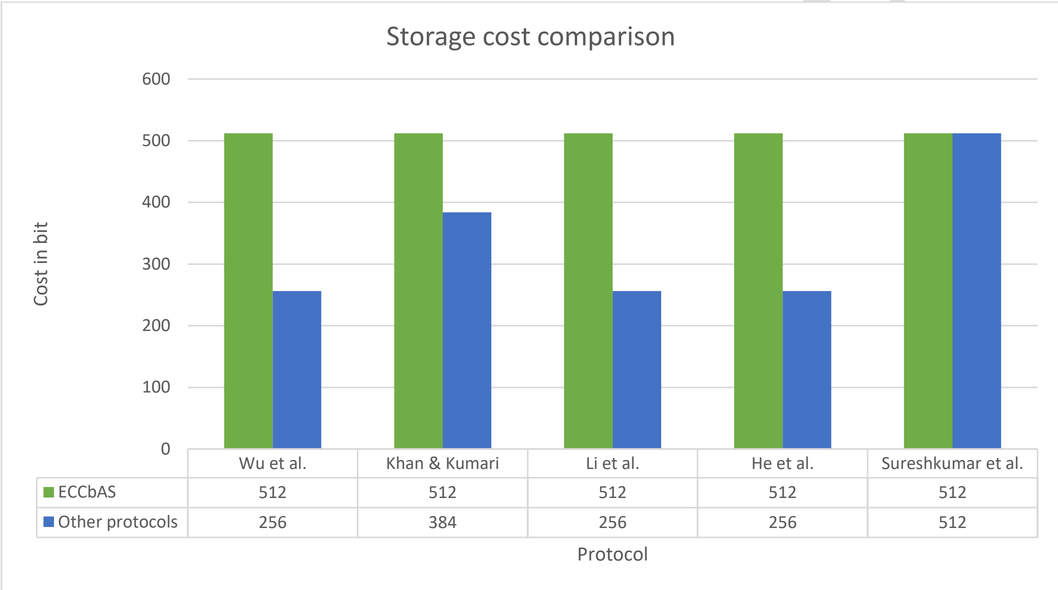


Figure 15: Comparison of storage cost of ECCbAS with recent similar protocols

presented in this paper.
Finally, analyzing security protocols advances the science of security protocol design while also raising awareness of the scenarios of these attacks in order to prevent such attacks in security protocol design.

References

Bahae Abidi, Abdelillah Jilbab, and Mohamed EL Haziti. Wireless sensor networks in biomedical: Wireless body area networks. In *Europe and MENA cooperation advances in information and communication technologies*, pages 321–329. Springer, 2017.

Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of Things: a survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4):2347–2376, 2015.

Zeeshan Ali, Anwar Ghani, Imran Khan, Shehzad Ashraf Chaudhry, SK Hafizul Islam, and Debasis Giri. A robust authentication and access control protocol for securing wireless healthcare sensor networks. *Journal of Information Security and Applications*, 52:102502, 2020. ISSN 2214-2126. doi: https://doi.org/10.1016/j.jisa.2020.102502. URL `https://www.sciencedirect.com/science/article/pii/S2214212619308361`.

Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. Federated learning for healthcare: Systematic review and architecture proposal. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–23, 2022.

Alessandro Armando, David Basin, Yohan Boichut, Yannick Chevalier, Luca Compagna, Jorge Cuéllar, P Hankes Drielsma, Pierre-Cyrille Héam, Olga Kouchnarenko, Jacopo Mantovani, et al. The AVISPA tool for the automated validation of internet security protocols and applications. In *International conference on computer aided verification*, pages 281–285. Springer, 2005.

Hamed Arshad and Abbas Rasoolzadegan. Design of a secure authentication and key agreement scheme preserving user privacy usable in telecare medicine information systems. *Journal of medical systems*, 40(11):1–19, 2016.

Atakan Arslan and Muhammed Ali Bingöl. Security and privacy analysis of recently proposed ECC-based RFID authentication schemes. *Cryptology ePrint Archive*, 2022.

Atakan Arslan, Sultan Aldırmaz Çolak, and Sarp Ertürk. A secure and privacy friendly ECC based RFID authentication protocol for practical applications. *Wireless Personal Communications*, 120(4):2653–2691, 2021.

Luigi Atzori, Antonio Iera, and Giacomo Morabito. The Internet of Things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

Paolo Bellavista, Luca Foschini, and Alessio Mora. Decentralised learning in federated deployment environments: A system-level survey. *ACM Computing Surveys (CSUR)*, 54(1):1–38, 2021.

Sravani Challa, Ashok Kumar Das, Vanga Odelu, Neeraj Kumar, Saru Kumari, Muhammad Khurram Khan, and Athanasios V. Vasilakos. An efficient ECC-based provably secure three-factor user authentication and key agreement protocol for wireless healthcare sensor networks. *Computers Electrical Engineering*, 69:534–554, 2018. ISSN 0045-7906. doi: https://doi.org/10.1016/j.compeleceng.2017.08.003. URL https://www.sciencedirect.com/science/article/pii/S0045790616302622.

Shehzad Ashraf Chaudhry, Khalid Yahya, Fadi Al-Turjman, and Ming-Hour Yang. A secure and reliable device access control scheme for IoT based sensor cloud systems. *IEEE Access*, 8:139244–139254, 2020. doi: 10.1109/ACCESS.2020.3012121.

Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. Project adam: Building an efficient and scalable deep learning training system. In *11th USENIX symposium on operating systems design and implementation (OSDI 14)*, pages 571–582, 2014.

Cas JF Cremers. The Scyther tool: Verification, falsification, and analysis of security protocols. In *International conference on computer aided verification*, pages 414–418. Springer, 2008.

Ashok Kumar Das. A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. *Peer-to-peer Networking and Applications*, 9(1):223–244, 2016.

Ashok Kumar Das, Mohammad Wazid, Animi Reddy Yannam, Joel J. P. C. Rodrigues, and Youngho Park. Provably secure ecc-based device access control and key agreement protocol for iot environment. *IEEE Access*, 7:55382–55397, 2019. doi: 10.1109/ACCESS.2019.2912998.

Souhir Gabsi, Yassin Kortli, Vincent Beroulle, Yann Kieffer, Areej Alasiry, and Belgacem Hamdi. Novel ECC-based RFID mutual authentication protocol for emerging IoT applications. *IEEE Access*, 9:130895–130913, 2021. doi: 10.1109/ACCESS.2021.3112554.

Keke Gai, Kim-Kwang Raymond Choo, Meikang Qiu, and Liehuang Zhu. Privacy-preserving content-oriented wireless communication in Internet of Things. *IEEE Internet of Things Journal*, 5(4):3059–3067, 2018.

Brij Gupta, Dharma P Agrawal, and Shingo Yamaguchi. *Handbook of research on modern cryptographic solutions for computer and cyber security*. IGI global, 2016.

Brij B Gupta. *Computer and cyber security: principles, algorithm, applications, and perspectives*. CRC Press, 2018.

Badis Hammi, Achraf Fayad, Rida Khatoun, Sherali Zeadally, and Youcef Begriche. A lightweight ECC-based authentication scheme for Internet of Things (IoT). *IEEE Systems Journal*, 14(3):3440–3450, 2020.

Debiao He, Neeraj Kumar, Jianhua Chen, Cheng-Chi Lee, Naveen Chilamkurti, and Seng-Soo Yeo. Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks. *Multimedia Systems*, 21(1):49–60, 2015.

Debiao He, Sherali Zeadally, Neeraj Kumar, and Jong-Hyouk Lee. Anonymous authentication for wireless body area networks with provable security. *IEEE Systems Journal*, 11:2590–2601, 2017.

Eric Horvitz and Deirdre Mulligan. Data, privacy, and the greater good. *Science*, 349(6245):253–255, 2015.

Xiaoying Jia, Debiao He, Neeraj Kumar, and Kim-Kwang Raymond Choo. Authenticated key agreement scheme for fog-driven IoT healthcare system. *Wireless Networks*, 25(8):4737–4750, 2019.

Qi Jiang, Jianfeng Ma, Xiang Lu, and Youliang Tian. An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. *Peer-to-peer Networking and Applications*, 8(6):1070–1081, 2015.

Zhanpeng Jin and Yu Chen. Telemedicine in the cloud era: Prospects and challenges. *IEEE Pervasive Computing*, 14(1):54–61, 2015.

Sonam Devgan Kaul and Amit K Awasthi. Security enhancement of an improved remote user authentication scheme with key agreement. *Wireless Personal Communications*, 89(2):621–637, 2016.

Salomeh Keyhani, Paul L Hebert, Joseph S Ross, Alex Federman, Carolyn W Zhu, and Albert L Siu. Electronic health record components and the quality of care. *Medical care*, pages 1267–1272, 2008.

Muhammad Khurram Khan and Saru Kumari. An improved user authentication protocol for healthcare services via wireless medical sensor networks. *International Journal of Distributed Sensor Networks*, 10(4):347169, 2014.

Jennifer King, Vaishali Patel, Eric W Jamoom, and Michael F Furukawa. Clinical benefits of electronic health record use: national findings. *Health services research*, 49(1pt2):392–404, 2014.

Djamel Eddine Kouicem, Abdelmadjid Bouabdallah, and Hicham Lakhlef. Internet of Things Security: a top-down survey. *Computer Networks*, 141:199–221, 2018.

Vikas Kumar, Rahul Kumar, Akber Ali Khan, Vinod Kumar, Yu-Chi Chen, and Chin-Chieh Chang. RAFI: robust authentication framework for IoT-based RFID infrastructure. *Sensors*, 22(9):3110, 2022.

Vinod Kumar, Musheer Ahmad, Dheerendra Mishra, Saru Kumari, and Muhammad Khurram Khan. RSEAP: RFID based secure and efficient authentication protocol for vehicular cloud computing. *Vehicular Communications*, 22:100213, 2020.

Saru Kumari, Muhammad Khurram Khan, and Xiong Li. An improved remote user authentication scheme with key agreement. *Computers & Electrical Engineering*, 40(6):1997–2012, 2014.

Cheng-Chi Lee, Chun-Ta Li, and Rui-Xiang Chang. A simple and efficient authentication scheme for mobile satellite communication systems. *International Journal of Satellite Communications and Networking*, 30(1):29–38, 2012.

Li Li, Yuxi Fan, Mike Tse, and Kuo-Yi Lin. A review of applications in federated learning. *Computers & Industrial Engineering*, 149:106854, 2020.

Xiong Li, Yongping Xiong, Jian Ma, and Wendong Wang. An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards. *Journal of Network and Computer Applications*, 35(2):763–769, 2012.

Xiong Li, Jian Ma, Wendong Wang, Yongping Xiong, and Junsong Zhang. A novel smart card and dynamic ID based remote user authentication scheme for multi-server environments. *Mathematical and Computer Modelling*, 58(1-2):85–95, 2013.

Xiong Li, Jianwei Niu, Saru Kumari, Junguo Liao, and Wei Liang. An enhancement of a smart card authentication scheme for multi-server architecture. *Wireless Personal Communications*, 80(1):175–192, 2015.

Xiong Li, Jianwei Niu, Saru Kumari, Junguo Liao, Wei Liang, and Muhammad Khurram Khan. A new authentication protocol for healthcare applications using wireless medical sensor networks with user anonymity. *Security and Communication Networks*, 9(15):2643–2655, 2016.

Chia-Hui Liu and Yu-Fang Chung. Secure user authentication scheme for wireless healthcare sensor networks. *Computers Electrical Engineering*, 59:250–261, 2017. ISSN 0045-7906. doi: https://doi.org/10.1016/j.compeleceng.2016.01.002. URL https://www.sciencedirect.com/science/article/pii/S0045790616000045.

Gavin Lowe. A hierarchy of authentication specifications. In *Proceedings 10th Computer Security Foundations Workshop*, pages 31–43. IEEE, 1997.

Azath Mubarakali. An efficient authentication scheme using blockchain technology for wireless sensor networks. *Wireless Personal Communications*, pages 1–15, 2021.

Bhawna Narwal and Amar Kumar Mohapatra. A survey on security and authentication in wireless body area networks. *Journal of Systems Architecture*, 113:101883, 2021.

Ben Niu, Xiaoyan Zhu, Haotian Chi, and Hui Li. Privacy and authentication protocol for mobile RFID systems. *Wireless Personal Communications*, 77(3):1713–1731, 2014.

Arezou Ostad-Sharif, Dariush Abbasinezhad-Mood, and Morteza Nikooghadam. A robust and efficient ECC-based mutual authentication and session key generation scheme for healthcare applications. *Journal of medical systems*, 43(1):1–22, 2019.

Quan Qian, Yan-Long Jia, and Rui Zhang. A lightweight RFID security protocol based on elliptic curve crytography. *Int. J. Netw. Secur.*, 18(2):354–361, 2016.

Minahil Rana, Akasha Shafiq, Izwa Altaf, Mamoun Alazab, Khalid Mahmood, Shehzad Ashraf Chaudhry, and Yousaf Bin Zikria. A secure and lightweight authentication scheme for next generation IoT infrastructure. *Computer Communications*, 165:85–96, 2021.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):1–7, 2020.

Samad Rostampour, Nasour Bagheri, Ygal Bendavid, Masoumeh Safkhani, Saru Kumari, and Joel JPC Rodrigues. An authentication protocol for next generation of constrained IoT systems. *IEEE Internet of Things Journal*, 2022.

Masoumeh Safkhani, Carmen Camara, Pedro Peris-Lopez, and Nasour Bagheri. RSEAP2: an enhanced version of RSEAP, an RFID based authentication protocol for vehicular cloud computing. *Vehicular Communications*, 28:100311, 2021.

Fatty M Salem and Ruhul Amin. A privacy-preserving RFID authentication protocol based on El-Gamal cryptosystem for secure TMIS. *Information Sciences*, 527:382–393, 2020.

K. Sowjanya, Mou Dasgupta, and Sangram Ray. Elliptic curve cryptography based authentication scheme for internet of medical things. *Journal of Information Security and Applications*, 58:102761, 2021. ISSN 2214-2126. doi: https://doi.org/10.1016/j.jisa.2021.102761. URL https://www.sciencedirect.com/science/article/pii/S2214212621000120.

Jangirala Srinivas, Dheerendra Mishra, and Sourav Mukhopadhyay. A mutual authentication framework for wireless medical sensor networks. *Journal of medical systems*, 41(5):1–19, 2017.

Christos Stergiou, Kostas E Psannis, Byung-Gyu Kim, and Brij Gupta. Secure integration of IoT and cloud computing. *Future Generation Computer Systems*, 78:964–975, 2018.

Venkatasamy Sureshkumar, Ruhul Amin, VR Vijaykumar, and S Raja Sekar. Robust secure communication protocol for smart healthcare system with FPGA implementation. *Future Generation Computer Systems*, 100:938–951, 2019.

Guo-heng Wei, Yan-lin Qin, and Wei Fu. An improved security authentication protocol for lightweight RFID based on ECC. *Journal of Sensors*, 2022, 2022.

Fan Wu, Lili Xu, Saru Kumari, and Xiong Li. An improved and anonymous two-factor authentication protocol for health-care applications with wireless medical sensor networks. *Multimedia Systems*, 23(2):195–205, 2017.

Kaiping Xue, Changsha Ma, Peilin Hong, and Rong Ding. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *Journal of Network and Computer Applications*, 36(1):316–323, 2013.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Fadila Zerka, Samir Barakat, Sean Walsh, Marta Bogowicz, Ralph TH Leijenaar, Arthur Jochems, Benjamin Miraglio, David Townend, and Philippe Lambin. Systematic review of privacy-preserving distributed machine learning from federated databases in health care. *JCO clinical cancer informatics*, 4:184–200, 2020.

Mohammad Reza Servati received his B.Sc. in Information Technology from the computer engineering department of Islamic Azad University in 2019. He is currently pursuing his M.Sc.'s at Shahid Rajaee Teacher Training University, Tehran, Iran. He is interested in the security of constraint environments such as IoT, WSN, MWSN, and other similar technologies.

Masoumeh Safkhani received the Ph.D. degree in Electrical Engineering from the Iran University of Science and Technology, in 2012, with a focus on security analysis of RFID protocols. She is currently an Associate Professor with the Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran. She is the author/coauthor of over 70 technical articles in information security and cryptology in major international journals and conferences. Her current research interests include security analysis of lightweight and ultra-lightweight protocols, targeting constrained environments, such as RFID, the IoT, VANET, and WSN.

	**Mohammad Reza Servati** received his B.Sc. in Information Technology from Islamic Azad University's computer engineering department in 2019. Shahid Rajaee Teacher Training University in Tehran is where he is currently working on his master's degree. Security of Constraints environments, such as IoT, cloud, and other similar technologies, are of interest to him.
	**Masoumeh Safkhani** received the Ph.D. degree in electrical engineering from the Iran University of Science and Technology, in 2012, with the security analysis of RFID protocols as her major field. She is currently an Assistant Professor with the Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran. Her current research interests include the security analysis of lightweight and ultra-lightweight protocols, targeting constrained environments, such as RFID, the IoT, VANET, and WSN. She is the author/coauthor of over 50

	technical articles in information security and cryptology in major international journals and conferences.

Declaration of interests

☒ 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.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: