Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.019595

ARTICLE

Amassing the Security: An Enhanced Authentication and Key Agreement Protocol for Remote Surgery in Healthcare Environment

Tsu-Yang Wu[1], **Qian Meng**[1], **Lei Yang**[1], **Saru Kumari**[2] and **Matin Pirouz**[3,*]

[1]Shandong University of Science and Technology, Qingdao, 266400, China

[2]Chaudhary Charan Singh University, Meerut, Uttar Pradesh, 250004, India

[3]California State University, Fresno, 93740, USA

*Corresponding Author: Matin Pirouz. Email: mpirouz@ieee.org

Received: 30 September 2021   Accepted: 09 March 2022

ABSTRACT

The development of the Internet of Things has facilitated the rapid development of various industries. With the improvement in people's living standards, people's health requirements are steadily improving. However, owing to the scarcity of medical and health care resources in some areas, the demand for remote surgery has gradually increased. In this paper, we investigate remote surgery in the healthcare environment. Surgeons can operate robotic arms to perform remote surgery for patients, which substantially facilitates successful surgeries and saves lives. Recently, Kamil et al. proposed a secure protocol for surgery in the healthcare environment. However, after cryptanalyzing their protocol, we deduced that their protocols are vulnerable to temporary value disclosure and insider attacks. Therefore, we design an improved authentication and key agreement protocol for remote surgeries in the healthcare environment. Accordingly, we adopt the real or random (ROR) model and an automatic verification tool Proverif to verify the security of our protocol. Via security analysis and performance comparison, it is confirmed that our protocol is a relatively secure protocol.

KEYWORDS

IoT; healthcare; security analysis; authentication; robotic arm; ROR

1 Introduction

As a novel paradigm, Internet of Things (IoT) [1–5] can effectively share data, coordinate and utilize resources. Simultaneously, in addition to reducing data transmission delay, the active of the emergence of the 5G [6] technology also improves the data transmission rate, which makes it possible to exchange of large amounts of data. This technology has been widely adopted in smart agriculture, smart cities, transportation, healthcare [7,8], artificial intelligence [9–11], etc., and has become an important part of people's life.

Healthcare is an important application of the IoT. With the improvement of living standards, the requirements for medical and health care are gradually increasing. Today, there is a substantial demand for medical and health care systems. The application of IoT in healthcare involves the use of the most advanced internet technology to realize interactions between patients and doctors and medical institutions and medical equipment, which enables the informatization. With the help of IoT technology, artificial intelligence [12] and intelligent equipment, we can build a perfect IoT medical system to solve or reduce the problems of difficult medical treatment and tense doctor-patient relationships caused by the lack of medical resources. Although healthcare can provide people with significant convenience, several security problems [13–17] exist, such as the disclosure of patients' medical data and the tampering of patients' medical schemes by illegal personnel of the system. Many researchers have proposed a large number of schemes [12,18–20] to address the security problems inherent in the healthcare environment. However, some existing authentication and key agreement protocols have security vulnerabilities, such as against offline guessing, impersonation and insider attacks. Therefore, it is crucial to propose an AKA protocol to address these challenges.

Wu et al. [21] proposed an authentication scheme, suitable for telemedicine information systems (TMIS). However, Debiao et al. [22] have confirmed that their scheme is vulnerable to several security problems, such as impersonation attacks and insider attacks. To address these vulnerabilities, Debiao et al. [22] proposed an improved scheme, which is also applicable to TMIS. Wei et al. [23] proposed a protocol suitable for TMIS without the pre-deployment phase; however, Zhu et al. [24] verified that the protocol proposed by Wei et al. [23] could not resist offline password guessing attacks. Xu et al. [25] proposed an elliptic curve cryptography (ECC)-based scheme. They claimed that their protocol can effectively provide authentication and user anonymity. However, Islam et al. [26] pointed out that Xu et al.'s [25] scheme are vulnerable to replay attacks and smart card stolen attacks, incorrect password update phase, and failure to successfully complete mutual authentication. Subsequently, Islam et al. [26] proposed an improved protocol based on the that proposed by Xu et al. [25]. The protocol was also designed based on ECC. Li et al. [27] designed an authentication scheme based on chaotic mapping; however, Madhusudhan et al. [28] proved that their scheme cannot successfully resist password guessing attacks. Zhang et al. [29] designed a three factor lightweight authentication agreement to address the problem of user anonymity in the e-healthcate system. However, Aghili et al. [30] pointed out that the agreement of Zhang et al. [29] cannot resist denial of service attacks (DOS) and insider attacks, as well as provide user untraceability and desynchronization. Therefore, Aghili et al. [30] proposed an improved scheme, which can provide user anonymity and mutual authentication. Sharma et al. [31] proposed a healthcare service authentication scheme based on cloud Internet of things, but Azrour et al. [32] pointed out that Sharma et al.'s [31] scheme could not resist user impersonatin attacks and offline password guessing attacks. Soni et al. [33] designed an authentication scheme for patient monitoring, but unfortunately, their scheme was proved by Xu et al. [34] that it could not provide perfect forward security. Kaur et al. [35] designed a secure protocol to solve the problem of security authentication in remote surgery. Ali et al. [36] designed a symmetric encryption and decryption scheme for TMIS; however, Yu et al. [37] discovered that this scheme [36] cannot withstand session key exposure attacks, man in the middle attacks (MITM) and impersonation attacks. Masud et al. [38] proposed a lightweight identity authentication scheme based on IoT healthcare. However, this scheme has been proved by Kwon et al. [39] that there are many security problems, such as offline password guessing, user impersonation, insider attacks and cannot ensure user anonymity. We summarize the literature reviewed in Table 1.

Influenced by COVID-19, the demand for remote surgery [40,41] under healthcare environment is gradually increasing. At the same time, the 5G network technology can transmit information with high efficiency and low delay, thereby facilitating remote surgery. The application of a remote surgery is shown in the Fig. 1. Surgeons can operate robotic arms to perform remote surgery for patients, which enables a number patients infected with the virus to receive prompt treatment, reduces the spread of the virus, and provide the stable development of society. Although the development of this technology can bring several benefits, they are highly dependent on the network, and there will be some security problems. For example, if network delay occurs when a surgeon remotely manipulates a robotic arm to operate a patient, the surgeon cannot obtain feedback information in time, which will adversely affect the operation process and severely endanger the patient's life. In addition, if an illegal surgeon manipulates the robotic arm or an unauthorized robotic arm is utilized, this will also threaten the safety of patients. Therefore, a secure lightweight authentication and key agreement protocol design is required to address these problems.

Table 1: Cryptographic techniques & limitations

Protocols	Cryptographic techniques	Limitations
Wu et al. [21]	(1) Utilized modular operation	(1) Cannot resist impersonation attacks
	(2) Utilized one-way hash function	Cannot resist insider attacks
Wei et al. [23]	(1) Utilized modular operation (2) Based on smart card	Cannot resist offline password guessing attacks
Xu et al. [25]	(1) Utilized ECC	(1) Cannot resist replay attacks
	(2) Based on a dynamic ID authentication	(2) Cannot resist smart card stolen attacks
	(3) Utilized one-way hash function (4) Based on smart card	(3) Cannot provide mutual authentication
Islam et al. [26]	(1) Based on anonymous authentication (2) Utilized one-way hash function (3) Based on smart card	Cannot resist user impersonation attacks
Li et al. [27]	(1) Based on chaotic mapping (2) Based on dynamic identity authentication (3) Based on smart card (4) Can resist impersonnation attacks	Cannot resist password guessing attacks
Zhang et al. [29]	(1) Based on smart card	(1) Cannot resist denial of service attacks (2) Cannot resist insider attacks
	(2) Based on dynamic identity authentication	(3) Cannot provide user untraceability (4) Cannot provide desynchronization
Sharma et al. [31]	Utilized one-way hash function	(1) Cannot resist offline password guessing attacks (2) Cannot resist user impersonation attacks

(Continued)

Table 1 (continued)

Protocols	Cryptographic techniques	Limitations
Soni et al. [33]	(1) Utilized one-way hash function (2) Utilized ECC	Cannot provide perfect forward security
Kaur et al. [35]	(1) Utilized one-way hash function (2) Utilized ECC	–
Ali et al. [36]	(1) Based on symmetric encryption (2) Based on smart card	(1) Cannot resist session key exposure attacks (2) Cannot resist man in the middle attacks (3) Cannot resist impersonation attacks
Masud et al. [38]	(1) Based on symmetric encryption (2) Based on smart card	(1) Cannot resist offline password guessing attacks (2) Cannot resist user impersonation attacks (3) Cannot resist insider attacks (4) Cannot ensure user anonymity
Kamil et al. [42]	(1) Utilized one-way hash function (2) Based on smart card	(1) Cannot resist insider attacks (2) Cannot resist temperory value leakege disclosure attacks

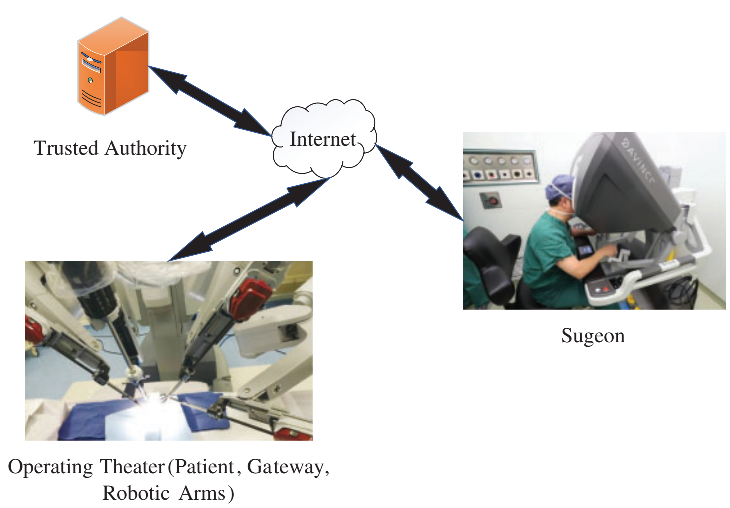

Figure 1: The application of a remote surgery

Recently, Kamil et al. [42] designed a lightweight authentication protocol that primarily solves identity authentication problem in remote surgery. Its remote surgery framework is illustrated in Fig. 2. This framework comprises four entities: a trusted authority (TA), surgeon, gateway, and robotic arm. All medical data during surgery is transmitted through tactile networks. To protect the security and privacy of medical data, the entire operation process needs to be completed under the detection of TA. Before surgery, surgeons and gateways, and the robotic arm must register with TA and obtain a legal identity. After each entity completes its registration, the surgeon, gateway, and robotic arm jointly decide on a session key to transmit data during surgery. They claim that their protocol is secure and efficient. However, we find that their protocol is vulnerable to temporary value disclosure attacks and insider attacks. In this paper, we propose an enhanced protocol suitable for this environment. Our contributions are: (1) We point out that Kamil et al.'s protocol has some security problems. (2) To solve these security problems, we propose an enhanced authentication protocol for remote surgery. Unlike Kamil et al.'s protocol, the registration phase of the robotic arm does not register with the TA via the gateway, because in an operating machine, the gateway and robotic arm are in the same system. We use ProVerif tool and ROR model to evaluate the security of the protocol. In addition, we use informal analysis to conduct a detailed security evaluation of the protocol, and prove that the protocol can resist common attacks, such as MIMT, replay attacks, impersonation attacks, insider attacks, etc. (3) Finally, through security and performance comparison, we find that our protocol is secure and suitable for the remote surgery environment.

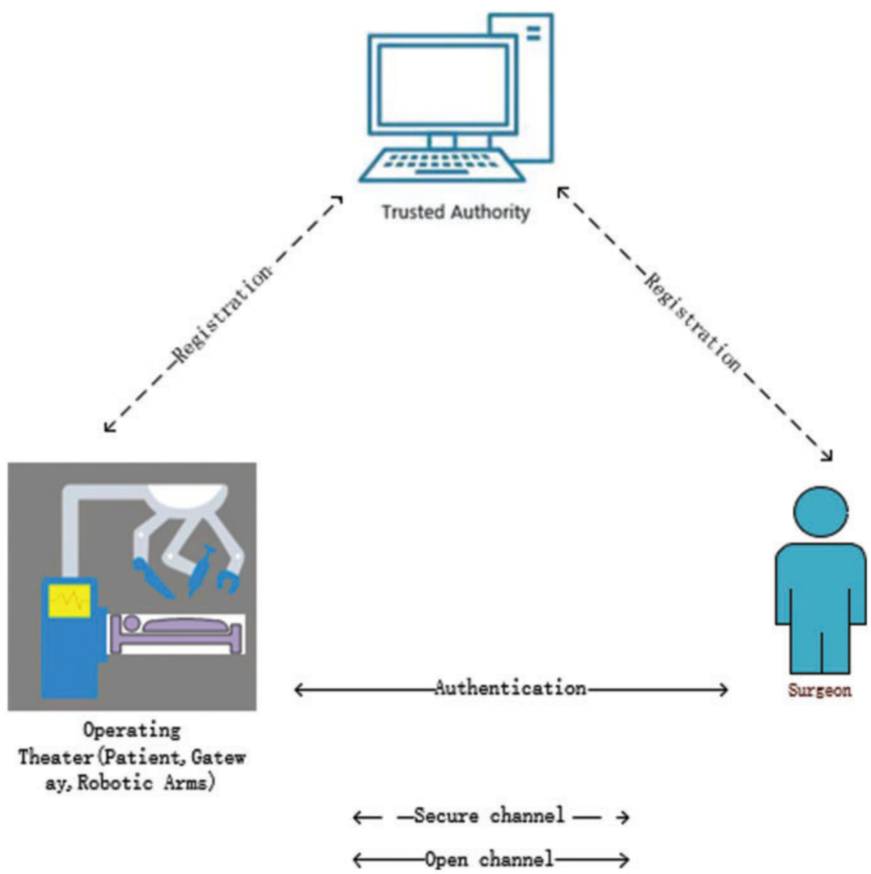

Figure 2: Network model

The remainder of this paper are arranged as follows. In Section 2, we review the protocol proposed by Kamil et al. The cryptanalysis of their protocol is then comprehensively introduced in detail in Section 3. In Section 4, we introduce our proposed protocol. Then, Section 5 presents a few security analyses of our protocol, while the performance comparison is introduced in Section 6. Finally, Section 7 concludes this paper.

2 Review of Kamil el at. Protocol

In this section, we review the protocol presented by Kamil et al. [42]. This protocol comprises seven phases; however, in this paper, we only adopt four phases: surgeon registration phase, gateway and robotic arm registration phase, user login, authentication and key agreement phase.

2.1 Surgeon Registration Phase

Surgeons are required to register with the TA as legitimate users to utilize robotic arms for remote surgeries. Messages at this stage are transmitted on a secure channel. The detailed steps are presented as follows in Table 2:

(1) S_i selects ID_i, PW_i, and a random number b_i, computes $D_i = h(ID_i \parallel b_i)$, $HPW_i = h(PW_i \parallel b_i)$, and then sends $\{D_i, HPW_i\}$ to TA.

(2) After receiving the message sent by S_i, TA selects a random number c_i, computes $\alpha = h(c_i \parallel D_k) \oplus h(D_i \parallel HPW_i)$, and $\beta = c_i \oplus h(ID_k \parallel D_k)$, stores $\{\alpha, \beta, h(\cdot)\}$ in the smart card (SC), and then sends SC to the user.

(3) After receiving SC, S_i computes $A_1 = h(PW_i \parallel ID_i) \oplus b_i$, $A_2 = h(b_i \parallel HPW_i \parallel D_i)$, and stores the $\{A_1, A_2\}$ in the SC.

Table 2: Notations and their meanings

Notations	Meanings
S_i	The i-th surgeon
ID_i	S_i's identity
PW_i	S_i's password
SC	The smart card
TA	The trusted authority
x	The secret key of TA
RM_j	The j-th robotic arm
ID_j	RM_j's identity
G_k	The k-th gateway
ID_k	G_k's identity
SK	Session-key
$h(\cdot)$	One way hash function
$Gen(\cdot), Rep(\cdot)$	Fuzzy extraction function

2.2 Gateway and Robotic Arm Registration Phase

At this phase, TA selects their respective identities for G_k and SN_j, computes some private parameters, and then transmits these private parameters to G_k and SN_j through secure channels. The detailed steps are presented as follows:

(1) TA selects its own identity ID_{TA}, a hash function $h(\cdot)$, and ID_j, ID_k, respectively, for the identity of G_k and SN_j, selects a random number s, computes $D_k = h(s \parallel ID_{TA} \parallel ID_k)$, $D_j = h(s \parallel ID_{TA} \parallel ID_j)$, and sends $\{ID_k, D_k, ID_j, D_j\}$ to the gateway.

2) After receiving the message sent by TA, G_k stores $\{ID_k, D_k, ID_j, D_j\}$ in its own memory, and then sends $\{ID_j, D_j\}$ to RM_j.

3) RM_j receives the message sent by G_k and stores $\{ID_j, D_j\}$ in its own memory.

2.3 Login and Authentication Phase

1) S_i inputs ID_i, PW_i, computes $b_i = A_1 \oplus h(PW_i \parallel ID_i)$, $D_i = h(ID_i \parallel b_i)$, $HPW_i = h(PW_i \parallel b_i)$, $A_2^* = h(b_i \parallel HPW_i \parallel D_i)$, and then performs authentication by checking $A_2^* \stackrel{?}{=} A_2$. If the authentication is successful, S_i selects a random number r_1 and timestamp T_1, and then computes $A_3 = \alpha \oplus h(D_i \parallel HPW_i)$, $A_4 = \beta \oplus T_1$, $A_5 = h(r_1 \parallel A_3 \parallel T_1)$, and $A_6 = (r_1 \parallel A_5) \oplus A_3$. After completing computation, it transfers the message $M_1 = \{A_4, A_5, A_6, T_1\}$ through the common channel to G_k.

2) After receiving the message M_1 sent by S_i, G_k first computes $c_i^* = A_4 \oplus h(ID_k \parallel D_k) \oplus T_1$, $A_3^* = h(c_i^* \parallel D_k)$, and $r_1^* \parallel A_5 = A_6 \oplus A_3$, and then verifies the timestamp $|T_k - T_1| \leqq \Delta T$ and $A_5^* \stackrel{?}{=} A_5$, where $A_5 = h(r_1 \parallel A_3^* \parallel T_1)$. If both are verified, G_k will select a random number r_2 and timestamp T_2, computes $A_7 = c_i \oplus h(ID_j \parallel D_j \parallel r_2 \parallel r_1^* \parallel T_2)$, $A_8 = D_j \oplus (r_2 \parallel r_1^* \parallel T_2)$, $A_9 = h(ID_j \parallel D_j \parallel c_i^* \parallel r_2 \parallel T_2)$, and then send the message $M_2 = \{A_7, A_8, A_9\}$ to RM_j through the commonchannel.

3) After receiving message M_2, RM_j first computes $r_2 \parallel r_1 \parallel T_2 = A_8 \oplus D_j$ and then verifies the timestamp $|T_R - T_2| \leqq \Delta T$. If the validation is successful, RM_j computes $c_i^{**} = A_7 \oplus h(ID_j \parallel D_j \parallel r_2^* \parallel r_1^{**} \parallel T_2)$, $A_9^* = h(ID_j \parallel D_j \parallel c_i^{**} \parallel r_2^* \parallel T_2)$ and checks $A_9^* \stackrel{?}{=} A_9$ to verify the identity of G_k. Subsequently, if the identification is successful, RM_j selects a random number r_2 and timestamp T_3, computes $K_1 = h(r_2^* \parallel r_1^{**} \parallel r_3)$, $A_{10} = h(r_2^* \parallel r_3 \parallel K_1 \parallel ID_j \parallel D_j \parallel T_3)$, $A_{11} = (r_3^* \parallel T_3) \oplus r_2$, and then sends message $M_3 = \{A_{10}, A_{11}\}$ to G_k through the common channel.

4) After receiving the message M_3, G_k computes $r_3^* \parallel T_3 = A_{11} \oplus r_2$ and verifies the timestamp $|T_k - T_3| \leqq \Delta T$. If the verification is successful, G_k computes the session key $K_2 = h(r_2 \parallel r_1^* \parallel r_3)$, then computes $A_{10}^* = h(r_2 \parallel r_3 \parallel K_2 \parallel ID_j \parallel D_j \parallel T_3)$, and verifies the correctness of the session key through $A_{10}^* \stackrel{?}{=} A_{10}$. After the successful verification, G_k selects the timestamp T_4, computes $A_{12} = h(K_2 \parallel r_2 \parallel r_3^* \parallel A_9 \parallel T_4)$, $A_{13} = (r_2 \parallel r_3^* \parallel T_4) \oplus r_1^*$, and then transmits the message $M_4 = \{A_8, A_{12}, A_{13}\}$ to S_i through the common channel.

5) After receiving the message M_4, S_i obtains the value of $r_2^* \parallel r_3^{**} \parallel T_4$ by computing $A_{13} \oplus r_1$, and then verifies the timestamp $|T_S - T_4| \leqq \Delta T$. If the verification is successful, S_i computes the session key $K_3 = h(r_2^* \parallel r_3^{**} \parallel r_1)$, $A_{12}^* = h(K_3 \parallel r_2^* \parallel r_3^{**} \parallel A_9 \parallel T_4)$, and verifies whether the session key is correct by checking $A_{12}^* \stackrel{?}{=} A_{12}$.

3 Cryptanalysis of Kamil et al.'s Protocol

In this section, based on the following attacker model [43], we analyze the security of the protocol proposed by Kamil et al. [42], and subsequently deduce that this protocol cannot resist temporary value disclosure attacks, insider attacks.

Attacker Model: Based on D-Y model [44], we define attacker $\mathcal{A}$ has the following capabilities:

1) $\mathcal{A}$ can block, steal, change and replay messages transmitted via a common channel, but a cannot obtain information transmitted via a secure channel;
2) $\mathcal{A}$ can steal the surgeon's smart card and extract the information stored in the smart card through power analysis;
3) $\mathcal{A}$ can be a malicious entity and can obtain the information stored in the gateway. $\mathcal{A}$ can also obtain the information stored in robotic arm's memory.

3.1 Insider Attacks

Insider attacks refers to a malicious person in the system who obtains the information stored in the system by other entities, uses the messages on the public channel, and finally successfully calculates the session key. Suppose a malicious attack $\mathcal{A}$ in the hospital obtains the content $\{ID_k, D_k, ID_j, D_j\}$ stored in the gateway during the registration phase, then he can launch the following attacks.

3.1.1 Impersonate the Surgeon

1) $\mathcal{A}$ obtains the message $\{ID_j, D_j\}$ stored in the gateway, and messages $M_1 = \{A_4, A_5, A_6, T_1\}$ and $M_2 = \{A_7, A_8, A_9\}$ on the common channel are also intercepted. Then, $\mathcal{A}$ can calculate $r_2 \parallel r_1 \parallel T_2 = A_8 \oplus D_j$, $c_i = A_7 \oplus h(ID_j \parallel D_j \parallel r_2 \parallel r_1^* \parallel T_2)$, $\beta = A_4 \oplus T_1$, and $A_3 = (r_1 \parallel A_5) \oplus A_6$.
2) $\mathcal{A}$ reselects a random number r_1' and timestamp T_1', then calculates $A_4' = \beta \oplus T_1'$, $A_5' = h(r_1' \parallel A_3 \parallel T_1')$, $A_6' = (r_1' \parallel A_5') \oplus A_3$, and then sends message $M_1' = \{A_4', A_5', A_6', T_1'\}$ to G_k.
3) After receiving message M_1', G_k calculates $c_i' = A_4' \oplus h(ID_k \parallel D_k) \oplus T_1'$, $A_3' = h(c_i' \parallel D_k)$, $r_1' \parallel A_5' = A_6' \oplus A_3'$. Subsequently, G_k checks the timestamp $|T_k - T_1'| \leq \Delta T$, if true, G_k verifies $A_5^* \stackrel{?}{=} A_5'$, where $A_5^* = h(r_1 \parallel A_3^* \parallel T_1)$. If the verification is successful, G_k selectes r_2, T_2, computes $A_7 = c_i' \oplus h(ID_j \parallel D_j \parallel r_2 \parallel r_1' \parallel T_2)$, $A_8 = D_j \oplus (r_2 \parallel r_1' \parallel T_2)$, $A_9 = h(ID_j \parallel D_j \parallel c_i' \parallel r_2 \parallel T_2)$, and then sends the message $M_2 = \{A_7, A_8, A_9\}$ to SN_j.
4) After SN_j receives M_2, it calculates $r_2 \parallel r_1' \parallel T_2 = A_8 \oplus D_j$, and then checks $|T_R - T_2| \leq \Delta T$. If true, SN_j verifies $A_9^* \stackrel{?}{=} A_9$, where $c_i' = A_7 \oplus h(ID_j \parallel D_j \parallel r_2 \parallel r_1' \parallel T_2)$, $A_9^* = h(ID_j \parallel D_j \parallel c_i' \parallel r_2 \parallel T_2)$. If the verification is successful, SN_j selects T_3, r_2, $K_1 = h(r_2 \parallel r_1' \parallel r_3)$, $A_{10} = h(r_2 \parallel r_3 \parallel K_1 \parallel ID_j \parallel D_j \parallel T_3)$, $A_{11} = (r_3 \parallel T_3) \oplus r_2$. Then it sends message $M_3 = \{A_{10}, A_{11}\}$ to G_k.
5) After receiving M_3, G_k calculates $r_3^* \parallel T_3 = A_{11} \oplus r_2$ and checks $|T_k - T_3| \leq \Delta T$; if true, it calculates $K_2 = h(r_2 \parallel r_1' \parallel r_3)$. G_k verifies $A_{10}^* \stackrel{?}{=} A_{10}$, where $A_{10}^* = h(r_2 \parallel r_3 \parallel K_2 \parallel ID_j \parallel D_j \parallel T_3)$. If the verification is successful, G_k selects T_4, calculates $A_{12} = h(K_2 \parallel r_2 \parallel r_3^* \parallel A_9 \parallel T_4)$, $A_{13} = (r_2 \parallel r_3^* \parallel T_4) \oplus r_1'$, and then sends $M_4 = \{A_8, A_{12}, A_{13}\}$ to S_i.
6) At this point, $\mathcal{A}$ intercepts the message M_4 sent by G_k and calculates $r_2^* \parallel r_3^* \parallel T_4 = A_{13} \oplus r_1'$, and the final session key $K = h(r_2^* \parallel r_3^* \parallel r_1')$.

3.1.2 Derive Session key

1. $\mathcal{A}$ intercepts the message $M_2 = \{A_7, A_8, A_9\}$ transmitted on the common channel. Accordingly, $\mathcal{A}$ can calculate $r_2 \parallel r_1 \parallel T_2 = A_8 \oplus D_j$.
2. After r_2 and r_1 are calculated, $\mathcal{A}$ intercepts the message $M_3 = \{A_{10}, A_{11}\}$ transmitted on the common channel, and then calculates $r_3^* \parallel T_3 = A_{11} \oplus r_2$. Therefore, $\mathcal{A}$ can calculate the session key $K_2 = h(r_2 \parallel r_1^* \parallel r_3)$.

In summary, we logically infer that the protocol proposed by Kamil et al. [42] cannot resist privileged insider attacks.

3.2 Temperory Value Disclosure Attacks

Assuming that attacker $\mathcal{A}$ obtains the random number r_1 selected by surgeon S_k in the login authentication phase, and intercepts the message A_{13} transmitted on the public channel, he can obtain the values of r_2^* and r_3^{**} by computing $A_{13} \oplus r_1$, and $\mathcal{A}$ can easily calculate the session key $K = h(r_2^* \parallel r_3^{**} \parallel r_1)$. Therefore, it can be concluded that their proposed protocol cannot resist the temporary value disclosure attacks.

4 The Proposed Protocol

In this section, we introduce the proposed protocol. The protocol comprises four phases: surgeon registration phase, gateway registration phase, robotic arm registration phase, login and authentication phase. Each phase will be comprehensively described in detail next.

4.1 Registration Phases

The registration phase mainly includes gateway registration, surgeon registration and robtic arm registration, which will be described in detail.

Surgeon Registration Phase: Before operating with a robotic arm, a surgeon must register with the TA as a legal user via a secure channel. Fig. 3 shows the surgeon's registration process. The specific steps necessary for this registration are as follows:

1) The surgeon S_i selects his own ID_i, PW_i, BIO_i, and a random number a_i, and then computes $Gen(BIO_i) = (\sigma_i, \tau_i)$, $RPW_i = h(PW_i \parallel a_i)$, $A_i = h(ID_i \parallel RPW_i \parallel \sigma_i)$, $TRPW_i = h(RPW_i \parallel \sigma_i)$. Subsequently, TA sends $\{ID_i, TRPW_i\}$ to TA.
2) After receiving the information sent by S_i, TA selects a random number b_i, and then computes $X = x \oplus h(b_i \parallel TRPW_i)$, $B_i = h(ID_i \parallel x) \oplus TRPW_i$, $D_i = b_i \oplus TRPW_i$. Subsequently, TA issues a smart card SC to the S_i, stores $\{B_i, D_i\}$ into the SC, and sends it to S_i.
3) After receiving the SC sent by TA, the surgeon stores $\{A_i, \tau_i\}$ in the SC.

Gateway Registration Phase: Before being utilized, the gateway must register with the TA and generate some private data for the authentication phase. Fig. 4 shows gateway's registration process. The specific steps required are as follows:

1. The gateway selects its own ID_k and sends it to the TA.
2. After receiving the message sent by the gateway, TA selects a random number d_k, computes $G_k = h(ID_k \parallel d_k)$, $G_x = G_k \oplus x$, and then sends G_k, d_k to the gateway.
3. Subsequently, the gateway stores G_k, d_k in its own memory.

S_i		TA
selects ID_i, PW_i, BIO_i		
$Gen(BIO_i) = (\sigma_i, \tau_i)$		
selects a_i		
$RPW_i = h(PW_i \parallel a_i)$		
$A_i = h(ID_i \parallel RPW_i \parallel \sigma_i)$		
$TRPW_i = h(RPW_i \parallel \sigma_i)$		
	$\xrightarrow{\{ID_i, TRPW_i\}}$	
		selects b_i
		$X = x \oplus h(b_i \parallel TRPW_i)$
		$B_i = h(ID_i \parallel x) \oplus TRPW_i$
		$D_i = b_i \oplus TRPW_i$
		stores $\{B_i, D_i\}$
		SC: $\{B_i, D_i, h(\cdot)\}$
	$\xleftarrow{\{SC\}}$	
Stores $\{A_i, \tau_i\}$ into SC		
SC: $\{a_i, A_i, \tau_i, B_i, D_i, h(\cdot)\}$		

Figure 3: Surgeon registration

G_k		TA
selects ID_k		
	$\xrightarrow{\{ID_k\}}$	
		Select d_k
		Compute $G_k = h(ID_k \parallel d_k)$
		$G_x = G_k \oplus x$
	$\xleftarrow{\{G_x, d_k\}}$	
Stores $\{G_x, d_k\}$ in its memory		

Figure 4: Gateway registration phase

Robotic Arm Registration Phase: Because the robotic arm and gateway are in the same system, the robotic arm is solely required to register with the gateway via a secure channel. Fig. 5 shows robotic arm's registration process. The specific steps required are comprehensively presented as follows:

1) The robotic arm RM_j selects its identity ID_j and sends it to the gateway via a secure channel.
2) After receiving a message sent by the robotic arm, gateway selects a random number c_j, and computes $x = h(ID_k \parallel d_k) \oplus G_x$, $E_j = h(ID_j \parallel x)$, $F_j = c_j \oplus E_j$; subsequently, G_k stores F_j and then sends $\{E_j, F_j\}$ to RM_j.
3) Finally RM_j saves $\{E_j, F_j\}$ in its memory.

RM_j		G_k
selects ID_j		
	$\xrightarrow{\{ID_j\}}$	
		Select c_j
		Compute $x = h(ID_k \parallel d_k) \oplus G_x$
		$E_j = h(ID_j \parallel x)$
		$F_j = c_j \oplus E_j$
	$\xleftarrow{\{E_j, F_j\}}$	
Stores $\{E_j, F_j\}$ in its memeory		

Figure 5: Robotic arm registration phase

4.2 Login and Authentication Phase

Before performing long-distance operations, surgeons need to manipulate robotic arms via an access gateway. After S_i logs into the system, G_k first verifies S_i's identity, and then sends an authentication request to RM_j. After RM_j completes the authentication, G_k sends an authentication message to S_i. After mutual authentication, the three entities establish a common session key for communications. The specific login authentication and session key establishment process are shown in Table 3 and comprehensively described as follows:

1) S_i inputes ID_i, PW_i, inprints BIO_i, and computes $\sigma_{i'} = Rep(BIO_i, \tau_i)$, $RPW_i = h(PW_i \parallel a_i)$, $A_i' = h(ID_i \parallel RPW_i \parallel \sigma_{i'})$, $A_{i'} = h(ID_i \parallel RPW_i \parallel \sigma_{i'})$, by checking $A_{i'} \stackrel{?}{=} A_i$ to verify whether the legality of S_i's identity. If the verification process is successful, S_i selects a random number r_1 and timestamp T_1, computes $TRPW_{i'} = h(RPW_i \parallel \sigma_{i'})$, $h(ID_i \parallel x) = B_i \oplus TRPW_{i'}$, $b_i = D_i \oplus TRPW_{i'}$, $x = X \oplus h(b_i \parallel TRPW_i)$, $C_1 = ID_i \oplus h(ID_k \parallel x)$, $C_2 = SID_j \oplus h(h(ID_i \parallel x) \parallel b_i)$, $C_3 = r_1 \oplus h(b_i \parallel ID_i)$, $C_4 = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$, and sends the message $M_1 = \{D_i, B_i, C_1, C_2, C_3, C_4, T_1\}$ to G_k.
2) After receiving the message M_1 sent by S_i, G_k first checks the timestamp $|T_1 - T_k| \leqq \Delta T$. If the verification is successful, it computes $x = h(ID_k \parallel d_k) \oplus G_x$, $ID_i = C_1 \oplus h(ID_k \parallel x)$, $TRPW_i = B_i \oplus h(ID_i \parallel x)$, $b_i = D_i \oplus TRPW_i$, $SID_j = C_2 \oplus h(h(ID_i \parallel x) \parallel b_i)$, $r_1 = C_3 \oplus h(b_i \parallel ID_i)$, $C_4' = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$, and checks $C_4' \stackrel{?}{=} C_4$ to verify S_i. If the verification passes, G_k selects the timestamp T_2 and random number r_1, computes $E_j = h(ID_j \parallel x)$, $c_j = F_j \oplus E_j$, $PID_j = h(ID_j \parallel c_j)$, $C_5 = r_2 \oplus PID_j$, $C_6 = h(PID_j \parallel r_2 \parallel c_j \parallel T_2)$, $C_7 = r_1 \oplus h(r_2 \parallel c_j)$, $C_8 = h(b_i \parallel c_j) \oplus h(ID_j \parallel r_2)$, and then sends the message $M_2 = \{C_5, C_6, C_7, C_8, T_2\}$ to SN_j.
3) After receiving the message M_2 sent by G_k, SN_j first checks the timestamp $|T_2 - T_j| \leqq \Delta T$. If the verification is successful, SN_j computes $c_j = F_j \oplus E_j$, $PID_j = h(ID_j \parallel c_j)$, $r_2 = C_5 \oplus PID_j$, $C_6{}' = h(PID_j \parallel r_2 \parallel c_j \parallel T_2)$, $C_6' \stackrel{?}{=} C_6$; if true G_k selects r_3, T_3, and verifies G_k's identity by computing $C_6' \stackrel{?}{=} C_6$. If this verification is successful, SN_j selects a random number r_3 and timestamp T_3, computes $r_1 = C_7 \oplus h(r_2 \parallel c_j)$, $h(b_i \parallel c_j) = C_8 \oplus h(ID_j \parallel r_2)$, $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j)$, $C_9 = h(SK \parallel h(b_i \parallel c_j) \parallel T_3)$, $C_{10} = r_3 \oplus h(r_1 \parallel ID_j)$, and then sends the message $M_3 = \{C_9, C_{10}, T_3\}$ to the gateway.
4) After receiving the message M_3 from SN_j, G_k first checks the timestamp $|T_3 - T_k| \leqq \Delta T$ and computes $r_3 = C_{10} \oplus h(r_1 \parallel ID_j)$, $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j)$, $C_9' = h(SK \parallel h(b_i \parallel c_j) \parallel T_3)$; subsequently, G_k verifies the identity of SN_j by calculating $C_9' \stackrel{?}{=} C_9$. After successful verification, G_k selects T_4, computes $C_{11} = r_2 \oplus h(TRPW_i \parallel r_1)$, $C_{12} = h(b_i \parallel c_j) \oplus h(b_i \parallel ID_i)$, $C_{13}' = h(SK \parallel r_2 \parallel r_3 \parallel T_4)$, and sends message M_4 to S_i.
5) When S_i receives the message from G_k, it first validates the timestamp $|T_4 - T_i| \leqq \Delta T$, then computes $r_3 = C_{10} \oplus h(r_1 \parallel ID_j)$, $r_2 = C_{11} \oplus h(TRPW_i \parallel r_1)$, $h(b_i \parallel c_j) = C_{12} \oplus h(b_i \parallel ID_i)$, $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$, $C_{13}' = h(SK \parallel r_2 \parallel r_3 \parallel T_4)$, and finally verifies $C_{13'} \stackrel{?}{=} C_{13}$. If the verification is successful, S_i saves SK for future communication.

Table 3: Login and authentication phase

D_i	G_k	SN_j
Inputs ID_i, PW_i, imprints BIO_i $\sigma_{i'} = Rep(BIO_i, \tau_i)$ $RPW_i = h(PW_i \parallel a_i)$ $A_{i'} = h(ID_i \parallel RPW_i \parallel \sigma_{i'})$		

(Continued)

Table 3 (continued)

D_i	G_k	SN_j
$Checks A_{i'} \stackrel{?}{=} A_i$. Selects r_1, T_1 $TRPW_{i'} = h(RPW_i \parallel \sigma_{i'})$ $h(ID_i \parallel x) = B_i \oplus TRPW_{i'}$ $b_i = D_i \oplus TRPW_{i'}$ $x = X \oplus h(b_i \parallel TRPW_i)$ $C_1 = ID_i \oplus h(ID_k \parallel x)$ $C_2 = ID_j \oplus h(h(ID_i \parallel x) \parallel b_i)$ $C_3 = r_1 \oplus h(b_i \parallel ID_i)$ $C_4 = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$ $\xrightarrow{M_1=\{D_i,B_i,C_1,C_2,C_3,C_4,T_1\}}$		
	Checks $\|T_1 - T_k\| \leqq \Delta T$ $x = h(ID_k \parallel d_k) \oplus G_x$ $ID_i = C_1 \oplus h(ID_k \parallel x)$ $TRPW_i = B_i \oplus h(ID_i \parallel x)$ $b_i = D_i \oplus TRPW_i$ $ID_j = C_2 \oplus h(h(ID_i \parallel x) \parallel b_i)$ $r_1 = C_3 \oplus h(b_i \parallel ID_i)$ $C_4' = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$ Checks $C_4' \stackrel{?}{=} C_4$. Selects r_1, T_2 $E_j = h(ID_j \parallel x),\ c_j = F_j \oplus E_j$ $PID_j = h(ID_j \parallel c_j),\ C_5 =$ $r_2 \oplus PID_j$ $C_6 = h(PID_j \parallel r_2 \parallel c_j \parallel T_2)$ $C_7 = r_1 \oplus h(r_2 \parallel c_j)$ $C_8 = h(b_i \parallel c_j) \oplus h(ID_j \parallel r_2)$ $\xrightarrow{M_2=\{C_5,C_6,C_7,C_8,T_2\}}$	
		Checks $\|T_2 - T_j\| \leqq \Delta T$ $c_j = F_j \oplus E_j,\ PID_j = h(ID_j \parallel c_j)$ $r_2 = C_5 \oplus PID_j,$ $C_6{}' = h(PID_j \parallel r_2 \parallel c_j \parallel T_2)$ Checks $C_6' \stackrel{?}{=} C_6$. Selects r_3, T_3 $r_1 = C_7 \oplus h(r_2 \parallel c_j)$ $h(b_i \parallel c_j) = C_8 \oplus h(ID_j \parallel r_2)$ $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j)$ $C_9 = h(SK \parallel h(b_i \parallel c_j) \parallel T_3)$ $C_{10} = r_3 \oplus h(r_1 \parallel ID_j)$ $\xrightarrow{M_3=\{C_9,C_{10},T_3\}}$
	Checks $\|T_3 - T_k\| \leqq \Delta T$ $r_3 = C_{10} \oplus h(r_1 \parallel ID_j)$	

(Continued)

Table 3 (continued)

D_i	G_k	SN_j
	$SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j)$ $C_9' = h(SK \parallel h(b_i \parallel c_j) \parallel T_3)$ Check $C_9' \stackrel{?}{=} C_9$. Selects T_4 $C_{11} = r_2 \oplus h(TRPW_i \parallel r_1)$ $C_{12} = h(b_i \parallel c_j) \oplus h(b_i \parallel ID_i)$ $C_{13}' = h(SK \parallel r_2 \parallel r_3 \parallel T_4)$ $\xrightarrow{M_4=\{C_{10},C_{11},C_{12},T_4\}}$	
Checks $\lvert T_4 - T_i \rvert \leqq \Delta T$ $r_3 = C_{10} \oplus h(r_1 \parallel ID_j)$ $r_2 = C_{11} \oplus h(TRPW_i \parallel r_1)$ $h(b_i \parallel c_j) = C_{12} \oplus h(b_i \parallel ID_i)$ $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$ $C_{13} = h(SK \parallel r_2 \parallel r_3 \parallel T_4)$ checks $C_{13}' \stackrel{?}{=} C_{13}$		

5 Security Analysis

In this section, we adopt Proverif, ROR model, and informal analysis to validate the security of our proposed protocol

5.1 Proverif

Four entities are adopted in our protocol: TA, G_k, S_i and RM_j. According to the registration and authentication processes of the four entities in the protocol, we utilize Proverif [45,46] to describe the entire protocol process, which is comprehensively presented below:

1) *ch* and *sch* are used to represent common channel and secure channel, respectively. The registration phase is carried out on the secure channel, while the login and authentication phase is conducted on the public channel. The session key adopts SK_i, SK_j, and SK_k to represent the session key of the surgeon, robotic arm, and gateway, respectively. We also define some operations, such as *hash*, *XOR*, etc. The defined query is adopted for security verification. The specific function definition is presented in Figs. 6a–6c.
2) S_i's process is illustrated in Fig. 7a.
3) G_k's process is presented in Fig. 7b.
4) R_j's process is illustrated in Fig. 7c.
5) TA's process is shown in Fig. 7d.
6) Fig. 6d presents the obtained verification results. The final results are "Query not attacker (SKi[]) is true," "Query not attacker (SKj[]) is true," "Query not attacker (SKk[])," "Query inj-event (SurgeonAuthed) ==> inj-event (SurgeonStarted) is true," "Query inj-event (RMA-cGateway) ==> inj-event(GatewayAcSurgeon) is true," "Query inj-event(GatewayAcRM)

==> inj-event(RMAcGateway) is true," and "Query inj-event(SurgeonAcGateway) ==> inj-event(GatewayAcRM) is true." Therefore, our protocol can successfully pass the security verification of Proverif and resist attacks.

```
(*  channel*)
free ch :channel. (*  public channel *)
free sch: channel [private]. (*  secure channel, used for
registering *)
(*  shared keys *)
free SKi : bitstring  [private].
free SKj : bitstring  [private].
free SKk : bitstring  [private].
free IDi : bitstring  [private].
(* constants  *)
free  x:bitstring [private].
(*  functions  &  reductions & equations  *)
fun h(bitstring) :bitstring. (*  hash function *)
fun mult(bitstring,bitstring) :bitstring. (* scalar
multiplication operation *)
fun add(bitstring,bitstring):bitstring. (* Addition operation *)
fun sub(bitstring,bitstring):bitstring. (* Subtraction
operation *)
fun mod(bitstring,bitstring):bitstring. (* modulus operation
*)
fun con(bitstring,bitstring):bitstring. (* concatenation
operation *)
reduc forall m:bitstring, n:bitstring; getmess(con(m,n))=m.
fun xor(bitstring,bitstring):bitstring. (*  XOR operation *)
equation forall m:bitstring, n:bitstring; xor(xor(m,n),n)=m.
fun Gen(bitstring):bitstring. (* Generator operation *)
fun Rep(bitstring,bitstring):bitstring.
```

(a) Definition

```
(*  queries  *)
query attacker(SKi).
query attacker(SKj).
query attacker(SKk).
query inj-event(SurgeonAuthed()) ==> inj-
event(SurgeonStarted()).
query inj-event(RMAcGateway()) ==> inj-
event(GatewayAcSurgeon()).
query inj-event(GatewayAcRM()) ==> inj-
event(RMAcGateway()).
query inj-event(SurgeonAcGateway()) ==> inj-
event(GatewayAcRM()).
(* event *)
event SurgeonStarted().
event SurgeonAuthed().
event GatewayAcSurgeon().
event RMAcGateway().
event GatewayAcRM().
event SurgeonAcGateway().
```

(b) Events

```
let ProcessTA= SurgeonReg  |   GatewayReg  .
(* --------- main---------- *)
process
(!ProcessSurgeon  |   !ProcessGateway
| !ProcessRoboticArm  )
```

(c) Main

Figure 6: Definitions and results

```
let ProcessSurgeon=
new IDi:bitstring;new PWi: bitstring;
new Bioi: bitstring;new ai:bitstring;
let (a: bitstring, b: bitstring)=Gen(Bioi) in
let RPWi=h(con(PWi,ai)) in
let Ai=h(con(con(IDi,RPWi),a)) in
let TRPWi=h(con(RPWi,a)) in
out(sch,(IDi, TRPWil));
in(sch,(xBi:bitstring,xDi:bitstring,xX:bitstring));
!(event SurgeonStarted();
let a=Rep(Bioi,b) in let RPWi=h(con(PWi,ai)) in
let Ai'=h(con(con(IDi,RPWi),a)) in
if Ai'=Ai   then new r1:bitstring;
new T1:bitstring;new IDk:bitstring;new IDj:bitstring;
let TRPWi=h(con(RPWi,a)) in
let g=xor(xBi,TRPWi) in let bi=xor(xDi,TRPWi) in
let x=xor(xX,h(con(bi,TRPWi))) in
let C1=xor(IDi,h(con(IDk,x))) in
let C2=xor(IDj,h(con(h(con(IDi,x)),i)))in
let C3=xor(r1,h(con(bi,IDi))) in
let C4=h(con(con(con(con(r1,IDi),IDj),bi),T1)) in
out(ch,(xBi,xDi,C1,C2,C3,C4,T1)); event
SurgeonAuthed();
in(ch,(xC10:bitstring,xC11:bitstring,xC12:bitstring,xC13:
bitstring,xT4:bitstring)); let
r3=xor(xC10,h(con(r1,IDij)))in
let r2=xor(xC1l1,h(con(TRPWi,r1))) in
let q=xor(xC12,h(con(bi,xDi))) in
let SKi=h(con(con(con(r1,r2),r3),q)) in
let C13'=h(con(con(con(SKi,r2),r3),xT4)) in
if C13'=xC13 then event SurgeonAcGateway(); 0).
```

(a) Surgeon's process

```
let ProcessGateway=
new IDk:bitstring;new IDj:bitstring;
out(sch,(IDk));in(sch,(yGx:bitstring,ydk:bitstring));
new cj:bitstring;
let x=xor(h(con(IDk,ydk)),yGx) in
let Ej=h(con(IDj,x)) in let Fj=xor(cj,Ej) in
!(in(ch,(yBi:bitstring,yDi:bitstring,yC1:bitstring,yC2:bitst
ring,
yC3:bitstring,yC4:bitstring,yT1:bitstring));let
x=xor(h(con(IDk,ydk)),yGx) in  let
IDi=xor(yC1,h(con(IDk,x))) in let
TRPWi=xor(yBi,h(con(IDi,x))) in
let bi=xor(yDi,TRPWi) in
let IDj=xor(yC2,h(con(h(con(IDi,x),bi)))) in
let r1=xor(yC3,h(con(bi,IDi))) in
let C4'=h(con(con(con(con(r1,IDi),IDj),bi),yT1)) in
if C4'=yC4 then event GatewayAcSurgeon();
new r2:bitstring;new T2:bitstring;
let Ej=h(con(IDi,x)) in let cj=xor(Fj,Ej) in
let PIDj=h(con(IDj,ci)) in let C5=xor(r2,PIDj) in
let C6=h(con(con(con(PIDj,r2),cj),T2)) in
let C7=xor(r1,h(con(r2,cj))) in let q=h(con(bi,cj)) in
 let C8=xor(q,h(con(IDj,r2))) in out(ch,(C5,C6,C7,C8,T2));
in (ch,(yc9:bitstring,yC10:bitstring,yT3:bitstring));
let r3=xor(yC10,h(con(r1,IDij))) in
let SKk=h(con(con(con(r1,r2),r3),q)) in
let c9'=h(con(con(SKk,q),yT3)) in
if C9'=yc9 then event GatewayAcRM();
new T4:bitstring;let C11=xor(r2,h(con(TRPWi,r1)))in
 let C12=xor(q,h(con(bi,yDi))) in
let C13=h(con(con(con(SKk,r2),r3),T4)) in  0).
```

(b) Gateway's process

```
let ProcessRoboticArm=
new IDj:bitstring;
out(sch,(IDj));out(sch,(IDj));
in(sch,(zEj:bitstring,zFj:bitstring));
!(in(ch,(zC5:bitstring,zC6:bitstring,ZC7:bitstri
ng,ZC8:bitstring,ZT2:bitstring));
let cj=xor(zFj,zEj) in
let PIDj=h(con(IDj,cj)) in let r2=xor(zC5,PIDij)
in let C6'=h(con(con(con(PIDj,r2),cj),zT2)) in
if C6'=zC6 then event RMAcGateway();
new r3:bitstring;new T3:bitstring;
let rl=xor(zC7,h(con(r2,cj))) in
let q=xor(zC8,h(con(IDij,r2))) in
let SKj=h(con(con(con(r1,r2),r3)))in
let C9=h(con(con(SKj,q),T3)) in
let C10=xor(r3,h(con(r1,IDj))) in
out(ch,(C9,C10,T3));0).
```

(c) RM's process

```
let SurgeonReg=
in(sch,(mIDi:bitstring,mTRPWi:bitstring));
new bi:bitstring;let
X=xor(x,h(con(bi,mTRPWi))) in
let g=h(con(mlIDi,x)) in
let Bi=xor(g,mTRPWi) in
let Di=xor(bi,mTRPWi) in
out(sch,(Bi,i));0.let GatewayReg=
in(sch,(mIDk:bitstring));new dk:bitstring;
let Gk=h(con(mIDk,dk)) in
let Gx=xor(Gk,x) in out(sch,(Gx,dk));
0.let ProcessTA= SurgeonReg | GatewayReg.
```

(d) TA's process

Figure 7: Process

5.2 Formal Security Analysis

In this section, we perform a security analysis on the proposed protocol in the ROR [19,47] model to demonstrate the protocol's security.

5.2.1 ROR Model

The proposed protocol contains four entities: a surgeon, gateway, TA, and robotic arm. In the ROR model, we adopt $\Pi^x_{D_i}$, $\Pi^y_{RM_j}$, $\Pi^z_{G_k}$, and Π^n_{TA} to denote the x-th doctor's instance, y-th robot arm instance, z-th gateway, and the n-th TA, respectively. We assume that attacker $\mathcal{A}$ can possess the following query capabilities: $Y = \Pi^x_{D_i}$, $\Pi^y_{RM_j}$, $\Pi^z_{G_k}$, and Π^n_{TA}.

$Execute(Y)$: If the attacker executes this query, it intercepts the messages transmitted between S_i, G_k and SN_j on the public channel. The specific query is shown in Table 4.

Table 4: Simulation of *Execute*

On a *Execute* query, we use the simulation of Send query to do the following operations: $Send(\Pi^x_{D_i}, start) \rightarrow (D_i, B_i, C_1, C_2, C_3, C_4, T_1)$, $Send(\Pi^z_{G_k}, (D_i, B_i, C_1, C_2, C_3, C_4, T_1)) \rightarrow (C_5, C_6, C_7, C_8, T_2)$, $Send(\Pi^y_{RM_j}, (C_5, C_6, C_7, C_8, T_2)) \rightarrow (C_9, C_{10}, T_3)$, $Send(\Pi^z_{G_k}, (C_9, C_{10}, T_3)) \rightarrow (C_{10}, C_{11}, C_{12}, T_4)$. This query is answered by $(D_i, B_i, C_1, C_2, C_3, C_4, T_1)$, $(C_5, C_6, C_7, C_8, T_2)$,, (C_9, C_{10}, T_3), and $(C_{10}, C_{11}, C_{12}, T_4)$.

$Send(Y, M)$: If the attacker executes this query, it sends the message M to Y, and can receive a response from Y. The specific query is shown in Table 5.

$Hash(string)$: If an attacker executes this query, it enters a string and gets its hash value. The specific query is shown in Table 6.

$Corrupt(Y)$: If an attacker executes this query, it obtains the private value of an entity, such as a long-term private key, a parameter stored in SC, or a temporary message. The specific query is shown in Table 6.

$Test(Y)$: If the attacker executes this query, it flips a coin c. If $c = 1$, $\mathcal{A}$ obtains the correct SK, and if $c = 0$, $\mathcal{A}$ obtains a string with an equal length to the SK. The specific query is shown in Table 6.

Table 5: Simulation of *Send* query

On a query $Send(\Pi^x_{D_i}, start)$, assuming that $\Pi^x_{D_i}$ is a normal state, we perform the following operations. Select r_{A1}, T_{A1}, and compute $TRPW_{i'} = h(RPW_i \parallel \sigma_{i'})$, $h(ID_i \parallel x) = B_i \oplus TRPW_{i'}$, $b_i = D_i \oplus TRPW_{i'}$, $x = X \oplus h(b_i \parallel TRPW_i)$, $C_1 = ID_i \oplus h(ID_k \parallel x)$, $C_2 = ID_j \oplus h(h(ID_i \parallel x) \parallel b_i)$, $C_3 = r_1 \oplus h(b_i \parallel ID_i)$, $C_4 = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$. Then, the query is answered by $M_1 = \{D_i, B_i, C_1, C_2, C_3, C_4, T_1\}$

(Continued)

Table 5 (continued)

On a query $Send(\Pi^z_{G_k}, (D_i, B_i, C_1, C_2, C_3, C_4, T_1))$, and assume that $\Pi_{G^z_k}$ is a normal state to perform the following operations. Compute x, ID_i, $TRPW_i$, b_i, ID_j, r_1, C_4, and check C_4, if equal, select r_{A2}, T_{A2}, and compute E_j, c_j, PID_j, C_5, C_6, C_7, C_8. Then, the query is answered by $M_2 = \{C_5, C_6, C_7, C_8, T_2\}$.
On a query $Send(\Pi^y_{RM_j}, (C_5, C_6, C_7, C_8, T_2))$, and assume that $\Pi^y_{RM_j}$, is a normal state to perform the following operations. Compute c_j, PID_j, r_2, C_6, and check C_6, if equal, select r_{A3}, T_{A3}, and compute r_1, $h(b_i \parallel c_j)$, SK, C_9, C_{10}. Then, the query is answered by $M_3 = \{C_9, C_{10}, T_3\}$. On a query $Send(\Pi^z_{G_k}, (C_9, C_{10}, T_3))$, and assume that $\Pi_{G^z_k}$, is a normal state to perform the following operations. Compute r_3, SK, C_9, and check C_9, if equal, select r_{A4}, T_{A4}, and compute C_{11}, C_{12}, C_{13}. Then, the query is answered by $M_4 = \{C_{10}, C_{11}, C_{12}, T_4\}$.
On a query $Send(\Pi^x_{D_i}, (C_{10}, C_{11}, C_{12}, T_4))$, and assume that $\Pi^x_{D_i}$, is a normal state to perform the following operations. Compute $r_3, r_2, h(b_i \parallel c_j), SK, C_{13}$, and check C_9, if equal, compute $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$. Otherwise, it will be terminated. Finally, the user instance accepts and terminates.

Table 6: Simulation of *Hash*, *Corrupt*, and *Test* query

For a record $(string, s)$ that appears in the $Hash(string)$ query, renturn $s = Hash(string)$. Otherwise, select an element s, add the record $(string, s)$ to the list, and return s.
On a query $Corrupt(\Pi^x_{D_i})$, and if $\Pi^x_{D_i}$ is accepted, the query is answered by the parameter $\{a_i, A_i, \tau_i, B_i.D_i, h(\cdot)\}$ in the smart card.
On a *Test* query, flip a coin c to get the result of SK. If $c = 1$, return SK; otherwise, return a string of the same length.

5.2.2 Theorem

In the ROR model, if $\mathcal{A}$ can execute the queries $Execute(Y)$, $Send(Y, M)$, $Hash(string)$, $Corrupt(Y)$, and $Test(Y)$, then the probability that the attacker can break the proposed protocol P in polynomial time is: $Adv^{\mathcal{P}}_{\mathcal{A}}(\xi) \leq q_{send}/2^{l-2} + 3q^2_{hash}/2^{l-1} + 2max\{C' \cdot q^{s'}_{send}, q_{send}/2^l\}$. Here, q_{send} denotes the number of queries executed; q_{hash} refers to the number of *Hash* executions; C' and s' are two constants, and l represents the bit length of the biological information [48].

5.2.3 Proof

We played five rounds of the game, $GM_i (i = 0, 1, 2, 3, 4)$. $Succ^{GM_i}_{A}(\xi)$ is denoted as the probability that $\mathcal{A}$ can win in GM_i. The detailed simulation steps of the query in the game are presented below.

GM_0: This game commences by flipping a coin c. GM_0 does not perform query; hence, we can obtain the probability that $\mathcal{A}$ can successfully break P as follows:

$$Adv^{\mathcal{P}}_{\mathcal{A}}(\xi) = |2Pr[Succ^{GM_0}_{\mathcal{A}}(\xi)] - 1|. \quad (1)$$

GM_1: GM_1 is an execute query added to GM_0. A can only intercept messages M_1, M_2, M_3, M_4 transmitted on the common channel in GM_1. Subsequently, $\mathcal{A}$ will obtain SK by $Test(Y)$ query;

however, r_1, r_2, r_3 cannot be obtained. Hence, the probability of GM_1 is equal to that of GM_0.

$$|Pr[Succ_{\mathcal{A}}^{GM_1}(\xi)]| = Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)]. \quad (2)$$

GM_2: GM_2 is based on GM_1 with the addition of Send query, and according to Zipf's law [48], we can obtain the probability of GM_2 as follows:

$$|Pr[Succ_{\mathcal{A}}^{GM_2}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_1}(\xi)]| \leq q_{send}/2^l. \quad (3)$$

GM_3: GM_3 is based on GM_2 with the *Hash* query added and the *Send* query removed. According to the birthday paradox, we can get the probability of GM_3 as:

$$|Pr[Succ_{\mathcal{A}}^{GM_3}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_2}(\xi)]| \leq q_{hash}^2/2^{l+1}. \quad (4)$$

GM_4: In GM_4, we analyze two events to verify the security of $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$. One is to verify perfect forward security by obtaining the long-term key x of TA, and the other is to obtain temporary information to verify that the protocol can resist temporary information disclosure attacks.

1) Perfect forward security: $\mathcal{A}$ adopts Π_{TA^n} to obtain the long-term key x of TA, or $\Pi_{D_i}^x$, $\Pi_{RM_j}^y$ or $\Pi_{G_k}^z$ to obtain the private value of the registration phase.
2) Temporary information disclosure attack: $\mathcal{A}$ adopts $\Pi_{D_i}^x$, $\Pi_{RM_j}^y$ or $\Pi_{G_k}^z$ to obtain the temporary information of the three parties.

For the first event, even if $\mathcal{A}$ gets the long-term key x of TA, or the private values of both in the registration phase, the random numbers r_1, r_2 and r_3 cannot be computed; hence, $\mathcal{A}$ cannot compute the value of SK, where $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$. For the second event, even if $\mathcal{A}$ can obtain r_1, the values of r_2, r_3, b_i, and c_j are kept secret; hence, SK cannot be computed. Similarly, even if $\mathcal{A}$ can obtain r_2 or r_3, the value of SK cannot be computed. Accordingly, we can obtain the probability of GM_4 as:

$$|Pr[Succ_{\mathcal{A}}^{GM_4}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_3}(\xi)]| \leq q_{send}/2^l + q_{hash}^2/2^{l+1}. \quad (5)$$

GM_5: In GM_5, $\mathcal{A}$ adopts $Corrupt(A)$ to query the smart card for parameters $\{a_i, A_i, \tau_i, B_i, D_i, h(\cdot)\}$ and we show that that the proposed protocol is resistant to offline key guessing attacks. S_i is registered using the password PW_i and biometric Bio_i. $\mathcal{A}$ attempts to guess $A_i = h(ID_i \parallel RPW_i \parallel \sigma_{i'})$; however, ID_i, RPW_i and σ_i are kept secret. The probability that $\mathcal{A}$ guesses bits of biological information is: $1/2^l$ [49]. In Zipf's law [48], when $q_{send} \leqq 10^6$, the probability that $\mathcal{A}$ can guess the password is greater than 0.5. Therefore, we can obtain the probability of GM_5 as:

$$|Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_4}(\xi)]| \leq max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\} \quad (6)$$

GM_6: In GM_6, to verify whether the protocol P can resist the impersonate attack, $\mathcal{A}$ queries $h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$, and the game is terminated. Hence, we can obtain the probability of GM_6 as:

$$|Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)]| \leq q_{hash}^2/2^{l+1}. \quad (7)$$

Because the probabilities of the success and failure of GM_6 are equal, the probability that $\mathcal{A}$ can guess the session key is:

$$Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)] = 1/2. \quad (8)$$

According to the above formula, we can obtain

$$
\begin{aligned}
1/2Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) &= |Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)] - 1/2| \\
&= |Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)]| \\
&= |Pr[Succ_{\mathcal{A}}^{GM_1}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)]| \\
&\le \sum_{i=0}^{5} |Pr[Succ_{\mathcal{A}}^{GM_{i+1}}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_i}(\xi)]| \\
&= q_{send}/2^{l-1} + 3q_{hash}^2/2^l + max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\}
\end{aligned}
\tag{9}
$$

Therefore, we can obtain

$$
Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) \le q_{send}/2^{l-2} + 3q_{hash}^2/2^{l-1} + 2max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\}. \tag{10}
$$

It is not difficult to infer that our protocol has successfully passed the security verification of ROR model, and that it can resist offline password guessing attacks, smart card stolen attacks, random number disclosure attacks, as well as provide perfect forward security.

5.3 Informal Security Analysis

In this section, we verify that our proposed protocol can resist some common attacks.

5.3.1 Impersonation Attacks

Attacker $\mathcal{A}$ is likely to impersonate any one of the surgeon, gateway, and sensor nodes.

1) Impersonate Surgeon: An attacker $\mathcal{A}$ can attempt to impersonate a surgeon by intercepting a message $M_1 = \{D_i, B_i, C_1, C_2, C_3, C_4, T_1\}$ on the public channel. He attempts to compute $C_1 = ID_i \oplus h(ID_k \parallel x)$, $C_2 = ID_j \oplus h(h(ID_i \parallel x) \parallel b_i)$, and $C_3 = r_1 \oplus h(b_i \parallel ID_i)$; however, $\mathcal{A}$ does not know the values of *x, bi*, and ID_i, Consequently he cannot compute the values of C_1, C_2, C_3, and C_4 accurately. So he cannot calculate to re-initiate a new message M_1'. Therefore, attacker $\mathcal{A}$ cannot impersonate a legitimate surgeon.
2) Impersonate gateway: An attacker $\mathcal{A}$ intercepts the message $M_2 = \{C_5, C_6, C_7, C_8, T_2\}$ transmitted on the common channel, tries to compute $PID_j = h(ID_j \parallel c_j)$, $C_6 = h(PID_j \parallel r_2 \parallel c_j \parallel T_2)$, $C_7 = r_1 \oplus h(r_2 \parallel c_j)$, $C_8 = h(b_i \parallel c_j) \oplus h(ID_j \parallel r_2)$, and change some of its values. However, because $\mathcal{A}$ cannot obtain the value of c_j, he cannot compute PID_j and r_2, and thus cannot correctly compute the value of C_6, therefore, they cannot re-initiate a message M_2', as well as impersonate a legitimate gateway.
3) Impersonate robotic arm: When an attacker $\mathcal{A}$ wants to impersonate a legitimate robotic arm, he does so by intercepting the message $M_3 = \{C_9, C_{10}, T_3\}$ on the common channel and tries to compute C_9, where $C_9 = h(SK \parallel h(b_i \parallel c_j) \parallel T_3)$ is the value for which gateway authenticates the RM_j, but he cannot compute to get the values of r_1, r_2 and $h(b_i \parallel c_j)$, so $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$ and C_9 cannot be computed. Therefore, attacker $\mathcal{A}$ cannot re-initiate a message $M_3{}'$, so he cannot successfully impersonate a legitimate robotic arm.

Therefore, we can conclude that our protocol can successfully resist impersonation Attacks.

5.3.2 Man-in-the-Middle Attacks

If an attacker $\mathcal{A}$ wants to launch a man-in-the-middle attack, he can do so by intercepting message $M_1 = \{D_i, B_i, C_1, C_2, C_3, C_4, T_1\}$ on the common channel and trying to turn M_1 into M_1' by changing the value of r_1 or T_1. But $\mathcal{A}$ does not know the values of messages $\{ID_i, TRPW_i, D_i\}$, so he cannot compute $b_i = D_i \oplus TRPW_i$, $x = h(ID_k \parallel d_k) \oplus G_x$, $C_2 = ID_j \oplus h(h(ID_i \parallel x) \parallel b_i)$, $C_3 = r_1 \oplus h(b_i \parallel ID_i)$, and $C_4 = h(r_1 \parallel ID_i \parallel ID_j \parallel b_i \parallel T_1)$. In this case, $\mathcal{A}$ also cannot compute and change M_2, M_3 and M_4, so our protocol can resist the man-in-the-middle attacks.

5.3.3 User Anonymity

Since no information about S_i's identity is directly stored in S_i's smart card, an attacker cannot obtain S_i's identity information through smart card stolen attacks. Moreover, although $\mathcal{A}$ can intercept the message $M_1 = \{D_i, B_i, C_1, C_2, C_3, C_4, T_1\}$ on the public channel, $\mathcal{A}$ does not know the values of x and ID_k; hence the attacker cannot obtain the ID_i of S_i by computing $ID_i = C_1 \oplus h(ID_k \parallel x)$. Therefore, our protocol can provide user anonymity.

5.3.4 Insider Attacks

We assume that attacker $\mathcal{A}$ obtains the information $\{G_x, d_k, F_j\}$ stored by the gateway in the registration phase, but since $\mathcal{A}$ does not know x, he cannot compute $c_j = F_j \oplus E_j$, $PID_j = h(ID_j \parallel c_j)$, and the values of r_1, r_2, r_3 are also unknown to $\mathcal{A}$, so $\mathcal{A}$ cannot compute the session key $SK = h(r_1 \parallel r_2 \parallel r_3 \parallel h(b_i \parallel c_j))$. Therefore, our protocol is resistant to insider attacks.

6 Security and Performance Comparison

In this section, we compare the security and performance with the protocols of Sharma et al. [31], Soni et al. [33], Kaur et al. [35], Masud et al. [38] and Kamil et al. [42], which are applicable to the healthcare environment. The detailed results of the comparison are comprehensively described in subsections.

6.1 Security Comparison

In this subsection, we compare the security of these five protocols. ✓ and × are used to indicate whether certain safety characteristics are satisfied. Implies that this characteristic is not considered. The comparison results are shown in Table 7. As can be seen from the table, Sharma et al. [31] protocol cannot resist user impersonation attacks and offline password guessing attacks. The protocol of Soni et al. [33] cannot provide perfect forward security. The protocol proposed by Masud et al. [38] cannot resist user impersonation attacks, offline password guessing attacks and insider attacks, and cannot provide user anonymity. The protocol of Kamil et al. [42] cannot resist insider attacks and temporary value disclosure attacks. The protocol in [35] and our protocol are secure.

Table 7: Comparisons of security

Security properties	[31]	[33]	[35]	[38]	[42]	Ours
Perfect forward secrecy	✓	×	✓	-	-	✓
Man-in-the-middle attacks	✓	✓	✓	✓	✓	✓
User anonymity	✓	✓	✓	×	✓	✓
Mutual authentication	✓	✓	✓	✓	✓	✓
User Impersonation attack	×	✓	✓	×	×	✓
Untraceability	✓	-	✓	✓	✓	✓
Replay attacks	✓	✓	✓	✓	✓	✓
Temporary value disclosure attacks	-	✓	✓	-	×	✓

(Continued)

Table 7 (continued)

Security properties	[31]	[33]	[35]	[38]	[42]	Ours
Off-line password guessing attacks	×	✓	✓	×	✓	✓
Insider attacks	✓	✓	✓	×	×	✓

6.2 Performance Comparison

Here, we compare the performance of these five protocols from two aspects: computional cost and communicational cost.

We adopted a computer with Windows10 operating system, Intel (R) core (TM) i5- 8500CPU@ 3.00 GHz 3.00 G processor, and 8 G memory. The development software we use was IntelliJ idea version 2019.3, which is based on the call of Java pairing library, signature library, and symmetric encryption/decryption function. We ran various operations on the computer 50 times, and then use the average value as the reference time for calculating the computional cost. In addition, we approximate the operation time of the fuzzy extractor to the calculation time of point multiplication, and the computational cost of XOR and join operations is negligible. Based on the results in Table 8, we can drive the comparative results of computational cost in Table 9 and Fig. 8 (original). The reason why the computational cost of protocols [33] and [35] is very high is that they both use point multiplication, and protocol a also uses symmetric encryption and decryption, which leads to great computational overhead. The reason why the computational cost of our protocol is higher than that of protocols [31,33] and [42] is that we use a fuzzy extractor, which occupy some computational overhead, and they only use one-way hash functions, resulting in slightly higher computational cost.

Table 8: The computational cost of complex operations

Operations	Symbolic	Total (ms)
Bilinear pairing	T_b	9.9
Point multiplication	T_m	12.3
Point addition	T_a	0.0580
Hash function	T_h	0.0052
Point exponentiation	T_e	10.3
Map to point hash function	T_{ph}	30.9
Symmetric encryption	T_{en}	4.7
Symmetric decryption	T_{de}	0.1347

Table 9: Comparative results of computational cost

Protocols	Surgeon/User	Gateway/Trusted authority	Robotic arm/Sensor	Tocal(ms)
Sharma et al. [31]	$11T_h$	$7T_h$	$12T_h$	0.156
Soni et al. [33]	$4T_m + 13T_h$	$5T_m + 9T_h$	$5T_h$	110.8404

(Continued)

Table 9 (continued)

Protocols	Surgeon/User	Gateway/Trusted authority	Robotic arm/Sensor	Tocal(ms)
Kaur et al. [35]	$4T_m + 6T_h + 2T_{en} + T_{de}$	$6T_m + 4T_h + 2T_{en} + 2T_{de}$	$3T_m + 2T_h + T_{en} + 2T_{de}$	184.1359
Masud et al. [38]	$3T_h$	$4T_h$	$2T_h$	0.048
kamil et al. [42]	$8T_h$	$8T_h$	$4T_h$	0.104
Ours	$T_m + 13T_h$	$19T_h$	$7T_h$	12.5028

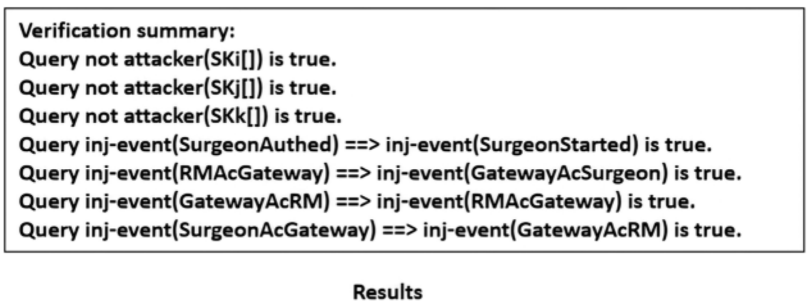

Figure 8: Results

For the communicational cost, we established that the output length of the single hash function H is 256 bits, T represents the timestamp, with a length of 32 bits, ID represents the length of the identity and is 256 bits, the length of encryption operation E is 256 bits, the length of group G is 1024 bits, and s represents the string with a length of 160 bits. According to the above definitions, Table 10 and Fig. 10 comprehensively show the results.

Table 10: Comparative results of communicational cost

Protocols	Communication costs (bits)	Length (bits)
Sharma et al. [31]	$9\|s\| + 7\|H\| + 5\|T\| + 2\|ID\|$	3648
Soni et al. [33]	$5\|s\| + 6\|H\| + 5\|T\| + 2\|G\|$	4544
Kaur et al. [35]	$3\|T\| + 4\|E\| + 3\|H\|$	1888
Masud et al. [38]	$9\|s\| + 4\|H\| + 3\|ID\|$	3232
Kamil et al. [42]	$6\|s\| + 4\|H\| + \|T\|$	2016
Ours	$12\|s\| + 3\|H\| + 3\|T\|$	2784

To sum up: Table 7 shows the comparison results of security. Table 9 and Fig. 9 are the comparison results of computational cost. Table 10 and Fig. 10 are the comparison results of communication cost. Although the computing cost of Sharma et al. [31] protocol is lower than ours, its security is not as good as ours, and the communication is also higher than ours; The protocols of Soni et al. [33] is not as good as our protocols in terms of security and performance; Although the protocol of Kaur et al. [35] is more secure and the communication cost is lower than ours, its computing cost is very high; Although the computational cost of Masud et al. [38] protocol is lower than ours, it has security problems and higher communication cost than ours; Although the protocol of Kamil et al. [42] has high performance and is better than ours, its security is worse than ours.

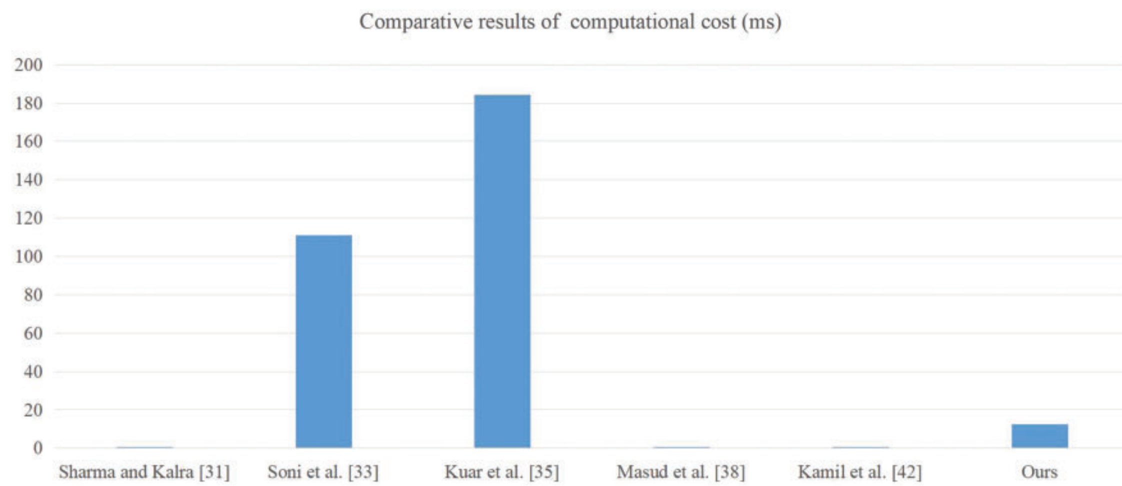

Figure 9: The comparison results of computational cost

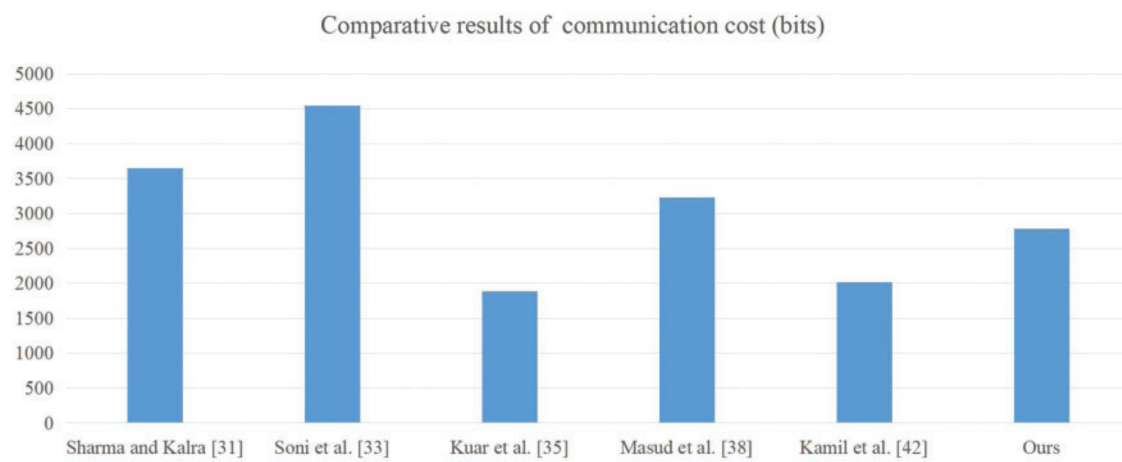

Figure 10: The comparison results of communication cost

7 Conclusion

In this paper, through the cryptanalysis of the protocol proposed by Kamil et al., we determined that their protocol cannot resist temporary value disclosure attacks and insider attacks. Then, we designed a novel authentication and key agreement protocol for remote surgeries in tactile network environments. We verified the security of our protocol via informal security analysis, and the ROR model and Proverif conducted formal security analysis on our protocol to further validate the security of the protocol. Finally, the performance comparison further indicates that our protocol is more suitable for tactile network environments. Furthermore, we hope that our research results will provide guidance for the development of intelligent medicine.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Xiong, H., Wu, Y., Jin, C., Kumari, S. (2020). Efficient and privacy-preserving authentication protocol for heterogeneous systems in Iot. *IEEE Internet of Things Journal, 7(12),* 11713–11724. DOI 10.1109/JIoT.6488907.
2. Xiong, H., Zhao, Y., Hou, Y., Huang, X., Jin, C. et al. (2020). Heterogeneous signcryption with equality test for iiot environment. *IEEE Internet of Things Journal, 8(21),* 16142–16152. DOI 10.1109/JIOT.2020.3008955.
3. Xue, X., Wu, X., Jiang, C., Mao, G., Zhu, H. (2021). Integrating sensor ontologies with global and local alignment extractions. *Wireless Communications and Mobile Computing, 2021,* 6625184. DOI 10.1155/2021/6625184.
4. Luo, Y., Weimin, Z., Chen, Y. C. (2021). An anonymous authentication and key exchange protocol in smart grid. *Journal of Network Intelligence, 6(2),* 2414–8105.
5. Wu, T. Y., Lee, Y. Q., Chen, C. M., Tian, Y., Al-Nabhan, N. A. (2021). An enhanced pairing-based authentication scheme for smart grid communications. *Journal of Ambient Intelligence and Humanized Computing,* 1–13. DOI 10.1007/s12652-020-02740-2.
6. Wu, T. Y., Lee, Z., Obaidat, M. S., Kumari, S., Kumar, S. et al. (2020). An authenticated key exchange protocol for multi-server architecture in 5G networks. *IEEE Access, 8,* 28096–28108. DOI 10.1109/Access.6287639.
7. Wu, J. M. T., Srivastava, G., Jolfaei, A., Fournier-Viger, P., Lin, J. C. W. (2021). Hiding sensitive information in ehealth datasets. *Future Generation Computer Systems, 117,* 169–180. DOI 10.1016/j.future.2020.11.026.
8. Wu, J. M. T., Tsai, M. H., Xiao, S. H., Liaw, Y. P. (2020). A deep neural network electrocardiogram analysis framework for left ventricular hypertrophy prediction. *Journal of Ambient Intelligence and Humanized Computing,* 1–17. DOI 10.1007/s12652-020-01826-1.
9. Meng, Z., Pan, J. S., Tseng, K. K. (2019). Pade: An enhanced differential evolution algorithm with novel control parameter adaptation schemes for numerical optimization. *Knowledge-Based Systems, 168,* 80–99. DOI 10.1016/j.knosys.2019.01.006.
10. Pan, J. S., Liu, N., Chu, S. C., Lai, T. (2021). An efficient surrogate-assisted hybrid optimization algorithm for expensive optimization problems. *Information Sciences, 561,* 304–325. DOI 10.1016/j.ins.2020.11.056.
11. Wu, J., Xu, M., Liu, F. F., Huang, M., Ma, L. et al. (2021). Solar wireless sensor network routing algorithm based on multi-objective particle swarm optimization. *Journal of Information Hiding and Multimedia Signal Processing, 12(1),* 1–11.
12. Xue, X., Zhang, J. (2021). Matching large-scale biomedical ontologies with central concept based partitioning algorithm and adaptive compact evolutionary algorithm. *Applied Soft Computing, 106,* 107343. DOI 10.1016/j.asoc.2021.107343.
13. Gritzalis, S., Lambrinoudakis, C., Lekkas, D., Deftereos, S. (2005). Technical guidelines for enhancing privacy and data protection in modern electronic medical environments. *IEEE Transactions on Information Technology in Biomedicine, 9(3),* 413–423. DOI 10.1109/TITB.2005.847498.
14. Pan, J. S., Sun, X. X., Chu, S. C., Abraham, A., Yan, B. (2021). Digital watermarking with improved SMS applied for QR code. *Engineering Applications of Artificial Intelligence, 97,* 104049. DOI 10.1016/j.engappai.2020.104049.
15. Zhang, Z., Chen, S., Sun, X., Liang, Y., Zhang, Z. et al. (2021). Trajectory privacy protection based on spatial-time constraints in mobile social networks. *Journal of Network Intelligence, 6(3),* 485–499.

16. Elshafey, M. A., Amein, A. S., Badran, K. S. (2021). Universal image steganography detection using multimodal deep learning framework. *Journal of Information Hiding and Multimedia Signal Processing, 12(3),* 152–161.
17. Chen, C. M., Deng, X., Kumar, S., Kumari, S., Islam, S. (2021). Blockchain-based medical data sharing schedule guaranteeing security of individual entities. *Journal of Ambient Intelligence and Humanized Computing,* 1–10. DOI 10.1007/s12652-021-03448-7.
18. Shamshad, S., Ayub, M. F., Mahmood, K., Kumari, S., Chaudhry, S. A. et al. (2021). An enhanced scheme for mutual authentication for healthcare services. *Digital Communications and Networks,* DOI 10.1016/j.dcan.2021.07.002.
19. Wu, T. Y., Wang, T., Lee, Y. Q., Zheng, W., Kumari, S. et al. (2021). Improved authenticated key agreement scheme for fog-driven IOT healthcare system. *Security and Communication Networks, 2021,* 6658041. DOI 10.1155/2021/6658041.
20. Wu, T. Y., Yang, L., Lee, Z., Chen, C. M., Pan, J. S. et al. (2021). Improved ecc-based three-factor multiserver authentication scheme. *Security and Communication Networks, 2021,* 6627956. DOI 10.1155/2021/6627956.
21. Wu, Z. Y., Lee, Y. C., Lai, F., Lee, H. C., Chung, Y. (2012). A secure authentication scheme for telecare medicine information systems. *Journal of Medical Systems, 36(3),* 1529–1535. DOI 10.1007/s10916-010-9614-9.
22. He, D. B., Chen, J. H., Zhang, R. (2012). A more secure authentication scheme for telecare medicine information systems. *Journal of Medical Systems, 36(3),* 1989–1995. DOI 10.1007/s10916-011-9658-5.
23. Wei, J., Hu, X., Liu, W. (2012). An improved authentication scheme for telecare medicine information systems. *Journal of Medical Systems, 36(6),* 3597–3604. DOI 10.1007/s10916-012-9835-1.
24. Zhu, Z. (2012). An efficient authentication scheme for telecare medicine information systems. *Journal of Medical Systems, 36(6),* 3833–3838. DOI 10.1007/s10916-012-9856-9.
25. Xu, X., Jin, Z. P., Zhang, H., Zhu, P. (2014). A dynamic ID-based authentication scheme based on ECC for telecare medicine information systems. *Applied Mechanics and Materials, 457,* 861–866. DOI 10.4028/AMM.457-458.861.
26. Islam, S. H., Khan, M. K. (2014). Cryptanalysis and improvement of authentication and key agreement protocols for telecare medicine information systems. *Journal of Medical Systems, 38(10),* 1–16. DOI 10.1007/s10916-014-0135-9.
27. Li, C. T., Lee, C. C., Weng, C. Y., Chen, S. J. (2016). A secure dynamic identity and chaotic maps based user authentication and key agreement scheme for e-healthcare systems. *Journal of Medical Systems, 40(11),* 1–10. DOI 10.1007/s10916-016-0586-2.
28. Madhusudhan, R., Nayak, C. S. (2019). A robust authentication scheme for telecare medical information systems. *Multimedia Tools and Applications, 78(11),* 15255–15273. DOI 10.1007/s11042-018-6884-6.
29. Zhang, L., Zhang, Y., Tang, S., Luo, H. (2017). Privacy protection for e-health systems by means of dynamic authentication and three-factor key agreement. *IEEE Transactions on Industrial Electronics, 65(3),* 2795–2805. DOI 10.1109/TIE.2017.2739683.
30. Aghili, S. F., Mala, H., Shojafar, M., Peris-Lopez, P. (2019). Laco: Lightweight three-factor authentication, access control and ownership transfer scheme for e-health systems in IOT. *Future Generation Computer Systems, 96,* 410–424. DOI 10.1016/j.future.2019.02.020.
31. Sharma, G., Kalra, S. (2019). A lightweight user authentication scheme for cloud-IOT based healthcare services. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 43(1),* 619–636. DOI 10.1007/s40998-018-0146-5.
32. Azrour, M., Mabrouki, J., Chaganti, R. (2021). New efficient and secured authentication protocol for remote healthcare systems in cloud-IOT. *Security and Communication Networks, 2021,* 5546334. DOI 10.1155/2021/5546334.

33. Soni, P., Pal, A. K., Islam, S. H. (2019). An improved three-factor authentication scheme for patient monitoring using WSN in remote health-care system. *Computer Methods and Programs in Biomedicine, 182,* 105054. DOI 10.1016/j.cmpb.2019.105054.
34. Xu, G., Wang, F., Zhang, M., Peng, J. (2020). Efficient and provably secure anonymous user authentication scheme for patient monitoring using wireless medical sensor networks. *IEEE Access, 8,* 47282–47294. DOI 10.1109/Access.6287639.
35. Kaur, K., Garg, S., Kaddoum, G., Guizani, M. (2020). Secure authentication and key agreement protocol for Tactile Internet-based tele-surgery ecosystem. *2020 IEEE International Conference on Communications (ICC)*, pp. 1–6. Dublin, Ireland. DOI 10.1109/ICC40277.2020.9148835.
36. Ali, Z., Hussain, S., Rehman, R. H. U., Munshi, A., Liaqat, M. et al. (2020). ITSSAKA-MS: An improved three-factor symmetric-key based secure AKA scheme for multi-server environments. *IEEE Access, 8,* 107993–108003. DOI 10.1109/ACCESS.2020.3000716.
37. Yu, S., Park, Y. (2020). Comments on "ITSSAKA-MS: An improved three-factor symmetric-key based secure AKA scheme for multi-server environments". *IEEE Access, 8,* 193375–193379. DOI 10.1109/ACCESS.2020.3032959.
38. Masud, M., Gaba, G. S., Choudhary, K., Hossain, M. S., Alhamid, M. F. et al. (2021). Lightweight and anonymity-preserving user authentication scheme for IOT-based healthcare. *IEEE Internet of Things Journal, 9,* 2649–2656. DOI 10.1109/JIOT.2021.3080461.
39. Kwon, D., Park, Y., Park, Y. (2021). Provably secure three-factor-based mutual authentication scheme with PUF for wireless medical sensor networks. *Sensors, 21(18),* 6039. DOI 10.3390/s21186039.
40. Anvari, M., Broderick, T., Stein, H., Chapman, T., Ghodoussi, M. et al. (2005). The impact of latency on surgical precision and task completion during robotic-assisted remote telepresence surgery. *Computer Aided Surgery, 10(2),* 93–99. DOI 10.3109/10929080500228654.
41. Wazid, M., Das, A. K., Lee, J. H. (2019). User authentication in a tactile internet based remote surgery environment: Security issues, challenges, and future research directions. *Pervasive and Mobile Computing, 54,* 71–85. DOI 10.1016/j.pmcj.2019.02.004.
42. Kamil, I. A., Ogundoyin, S. O. (2021). A lightweight mutual authentication and key agreement protocol for remote surgery application in tactile internet environment. *Computer Communications, 170,* 1–18. DOI 10.1016/j.comcom.2021.01.025.
43. Chaudhry, S. A. (2021). Combating identity de-synchronization: An improved lightweight symmetric key based authentication scheme for IOV. *Journal of Network Intelligence, 6(4),* 656–667.
44. Dolev, D., Yao, A. (1983). On the security of public key protocols. *IEEE Transactions on Information Theory, 29(2),* 198–208. DOI 10.1109/TIT.1983.1056650.
45. Blanchet, B. (2008). A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing, 5(4),* 193–207. DOI 10.1109/TDSC.2007.1005.
46. Abadi, M., Fournet, C. (2001). Mobile values, new names, and secure communication. *ACM Sigplan Notices, 36(3),* 104–115. DOI 10.1145/373243.360213.
47. Canetti, R., Goldreich, O., Halevi, S. (2004). The random oracle methodology, revisited. *Journal of the ACM, 51(4),* 557–594. DOI 10.1145/1008731.1008734.
48. Wang, D., Cheng, H., Wang, P., Huang, X., Jian, G. (2017). Zipf's law in passwords. *IEEE Transactions on Information Forensics and Security, 12(11),* 2776–2791. DOI 10.1109/TIFS.2017.2721359.
49. Odelu, V., Das, A. K., Goswami, A. (2015). A secure biometrics-based multi-server authentication protocol using smart cards. *IEEE Transactions on Information Forensics and Security, 10(9),* 1953–1966. DOI 10.1109/TIFS.2015.2439964.