Computer Communications 165 (2021) 85–96

Contents lists available at ScienceDirect

Computer Communications

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

A secure and lightweight authentication scheme for next generation IoT infrastructure

Minahil Rana [a], Akasha Shafiq [a], Izwa Altaf [a], Mamoun Alazab [b], Khalid Mahmood [a], Shehzad Ashraf Chaudhry [c], Yousaf Bin Zikria [d,*]

[a] Department of Computer Science, COMSATS University, Islamabad, Sahiwal Campus, Pakistan
[b] College of Engineering, IT and Environment, Charles Darwin University, 0810 NT, Darwin, Australia
[c] Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University, Istanbul, Turkey
[d] Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, South Korea

ARTICLE INFO

Keywords:
Authentication
6G/IoT security
Network Security
User impersonation

ABSTRACT

While the 6G/IoT transition is on the cards, the real advantage of this transition can be realized only if the user privacy and security are guaranteed. The smartcard and password based authentication protocols can help the transition in a rapid way. However, due to insecurities and/or heavy computation, many such protocols cannot cope with the dynamic requirements of future generation networks. Recently, Kaul and Awasthi presented a robust and secure user authentication protocol based on resource friendly symmetric cryptography primitives. They declared that their introduced protocol is convenient, efficient, and secure for the applications in real-world. In contrast, this article describes that protocol of Kaul and Awasthi is not secure because an attacker can easily find the identity of a legal user that is being sent on the public channel. Further, by using the identity of a legitimate user, an attacker can impersonate himself as a legitimate user of the system and can enjoy the services given by the server. So, their protocol is susceptible to user impersonation attacks, and their claim of being secure is proven to be wrong. Therefore, we have extended their work and presented an upgraded scheme by ensuring secure communication over the entire channel. Moreover, our proposed scheme is safe not solely against user impersonation attack but also major security attacks with reasonable communication, computation, and storage costs and is a better candidate for deployment in 6G/IoT networks.

1. Introduction

The 6G and Internet of Things (6G/IoT) are proposed to replace the existing communication infrastructure to provide endless connectivity. With an estimation of over fifty billion IoT devices till the end of the year 2020, the need for security and privacy for the users is growing. The users can take benefit the on demand infrastructure access in 6G/IoT revolution. However, the revolution comes with additional threats as compared with existing infrastructure, and the real benefit can only be realized after ensuring the security and privacy of the user. In the mechanism of smart card based distant user authentication, legal user and remote server authenticate each other on a transmission medium, which is not secure. The purpose of this mechanism is to provide on demand resources to legitimate service seekers remotely.

In 1981, Lamport [1] was the pioneer to introduce a remote user authenticated scheme on an insecure communication medium. This scheme was based on verification tables and passwords. Later on, it was identified that to ensure the safety needs of today's digital world, the dependence on the validation tables is inadequate. To guarantee the secure transmission, authentication protocols based on the smart card are presented by Hwang and Li [2] and Chang and Wu [3], in 2001 respectively. According to the user's concern, efficiency and security are the important parameters of authentication protocol. By keeping this user's view in mind, many distant user authentication schemes [4–10] were presented.

Das et al. [11], in 2004, introduced the idea of pseudo ID based distant user authenticated protocol by utilizing the smart card. Still, this scheme was not practical because it was vulnerable to numerous security attacks. Afterward, Liao et al. [12] carried the previous work and introduced a mechanism of mutual authentication with enhanced security features. However, in 2006, Yoon and Yoo [13] demonstrated various security flaws in Liao et al.'s [12] scheme. Thus, Wang et al. [14], in 2009, also introduced an improved scheme of Das et al. [11] with an enhancement of password authentication, that still has major features of the original scheme and resists their weaknesses.

After that, Wen and Li [15], in 2012, analyzed that Wang et al.'s protocol [14] does not combat user and server impersonation attacks.

* Corresponding author.
E-mail address: yousafbinzikria@ynu.ac.kr (Y.B. Zikria).

https://doi.org/10.1016/j.comcom.2020.11.002
Received 18 June 2020; Received in revised form 5 October 2020; Accepted 2 November 2020
Available online 5 November 2020

Moreover, the user's secret credentials can be leaked out by implementing an offline-password-guessing attack. Moreover, an insider, by using smart card parameters, can access all the secret factors of the legal user. Further, Chang et al. [16], in 2014, determined that Wang et al.'s [14] pseudo ID based scheme is insecure because the ID of a user is submitted in plaintext during the login phase. Besides, without any crucial verification, the adversary can exchange the user's password with a new password. Then, Chang et al. [16] introduced pseudo ID based authenticated protocol with the enhancement of an authoritative password update.

Lately, Kumari et al. [17] described that Chang et al.'s [16] protocol is vulnerable to impersonation, offline password guessing, insider, and the server masquerading attacks. Moreover, they highlighted the loopholes which are present in the phase of password change. Further, the protocol of [16] does not maintain a session key agreement to communicate in the future. Consequently, Kumari et al. [17] presented a modified scheme for the distant user authentication along with the key acknowledgment to reduce stated security vulnerabilities, also they declared that their protocol is more protected, efficient, and suitable for the applications used in real life. Chaudhry et al. [18] also explained the design faults of some previous schemes and proposed some measures for avoiding the design faults. Hussain et al. [19] also proposed some design measures for the authentication schemes proposed using only symmetric key functions. Some other relevant schemes were presented by various researchers [20–26]. Chen et al. also explained some of the attacks on password based schemes [27]. However, due to the usage of public key based operation, some of these schemes cannot be used in resource sensitive applications.

1.1. Motivations

Presently, Kaul and Awasthi [28] highlighted that Kumari et al.'s [17] proposed protocol is still vulnerable, as an attacker can easily get secure parameters of the scheme. The attacker also can obtain the session key, which is exchanged between the server and the user for future communication. In addition, the adversary can obtain the password of a legitimate user and server's private key. Due to this, the entire system collapses. Hence, Kaul and Awasthi [28] introduced a modified and efficient authentication scheme to get rid of stated security weaknesses in [17].

1.2. Contributions

Our paper highlights that Kaul and Awasthi's [28] scheme is susceptible to user masquerading attacks. An attacker can masquerade himself as a legitimate user and can steal secret parameters of the legitimate user. Thus, we have presented an improved and more secure distant user authenticated protocol to resists numerous security weaknesses.

1.3. Paper organization

The remaining paper is divided into eight sections, which are stated as: Preliminaries are demonstrated in Section 2. Kaul and Awasthi's [28] user authentication scheme is reviewed in Section 3. In Section 4, we have presented the cryptanalysis of Kaul and Awasthi's [28] scheme. The proposed scheme is described in Section 5. Formal and informal security analysis of our enhanced protocol is described in Section 6. Section 7 evaluates security and performance comparison. Finally, in Section 8, we have concluded the paper.

2. Preliminaries

In this section, there are explanations of basic notions that include adversarial model, symbols used, non-collisional hash function, and elliptic curve cryptography. Symbols that are used in this article are illustrated in Table 1.

2.1. Hash function

If non-collision hashing function H takes a random length string str as an input then it will generate fixed-length output code ($R = H(str)$). The output that is generated from the hash function is represented as a hash code or hash value. A minor change in an input $string$ can produce a major change in resultant output. A secure hash function must have the following characteristics:

- If input $string$ is defined, then it is computationally effortless to generate hash code $R = H(str)$.
- If $R = H(str)$ is defined, then it is computationally absurd and impractical to estimate the value of input $string$.
- It is an exhausting task to find definite inputs str_a and str_b like that $H(str_a) = H(str_b)$. Such specification is known as collision resistance.

Definition 1 (*Specifications of Collision Resistance*). Primarily arrange collision resistance of secure hashing function. The possibility that an attacker can find out a pair of strings ($str_a \neq str_b$) such that $H(str_a) = H(str_b)$ confined as $Adv_{\mathcal{A}}^{HASH}(t) = Prb[(str_a, str_b) \Leftarrow_r \mathcal{A} : (str_a \neq str_b)$ and $H(str_a) = H(str_b)]$, whereas adversary has favor to randomly select a pair (str_a, str_b). Attacker $\mathcal{A}$ can take benefit over the random selection as it can be computed in polynomial time. Whereas, collision resistance decides that $Adv_{\mathcal{A}}^{HASH}(t) \leq \in$, whereas $\in > 0$, is adequately a trivial number.

2.2. Adversarial/threat model

As declared in [29–37], the same threat model is acknowledged in this article in which according to the abilities of the attacker ($\mathcal{A}$), following steps are taken:

1. $\mathcal{A}$ has access over the full public transmission link. $\mathcal{A}$ has the ability to intercept, update, alter, drop, or send a duplicate message.
2. $\mathcal{A}$ can intercept credentials stored in SC by using power-analysis as stated in [38,39].
3. $\mathcal{A}$ can be anyone, such as a stranger, a legitimate server, or a legal user of the system.
4. IDs of legitimate user and server are announced publicly.
5. $\mathcal{A}$ cannot be able to launch an attack on the server as it is considered to be secured.

3. Review of the kaul and Awasthi's scheme

We have comprehensibly demonstrated distant user authenticated key agreement protocol presented by Kaul and Awasthi [28], in this section. Their scheme has four stages: registration, login, authentication, and password change.

3.1. Registration phase

The registration phase is shown in Fig. 1. In this phase, server S registers the user U_c by using these steps:

1 Initially, cth user U_c selects his ID_c, PW_c and random number m. Then, by calculating $RPW_c = h(m \parallel PW_c)$, U_c sends request $\{ID_c, RPW_c\}$ to S on a secure channel.
2 After that, the server generates an arbitrary number y_c for U_c to calculate the following values:

$$\alpha_c = h((ID_c \oplus a) \parallel b) \quad (1)$$

$$\beta_c = \alpha_c \oplus h(ID_c \oplus RPW_c) \quad (2)$$

$$\gamma_c = y_c \oplus h(\alpha_c \oplus RPW_c) \quad (3)$$

$$\chi_c = h(ID_c \| RPW_c \parallel y_c \| \alpha_c) \quad (4)$$

Table 1
Symbols.

Symbols	Detail	Symbols	Detail
$\mathcal{U}_c$	cth legal user	ID_c	cth user identity
PW_c	cth user password	$\mathcal{S}$	Legal server
a, b	Private key and number of server	y_c	Arbitrary number for $\mathcal{U}_c$
SC_c	User's Smart Card	T_1	Time stamp obtained at User's side
T_2	Server's current time stamp	$T^{'}$	Threshold value
δT_c	Time of transmission delay	$\|\|$	Concatenation operator
$\oplus$	XoR operator	$h(\cdot)$	Non-collision hash function
SK	Session key	$\mathcal{A}$	Adversary
$\Rightarrow$	Private communication channel	$\rightarrow$	Public communication channel

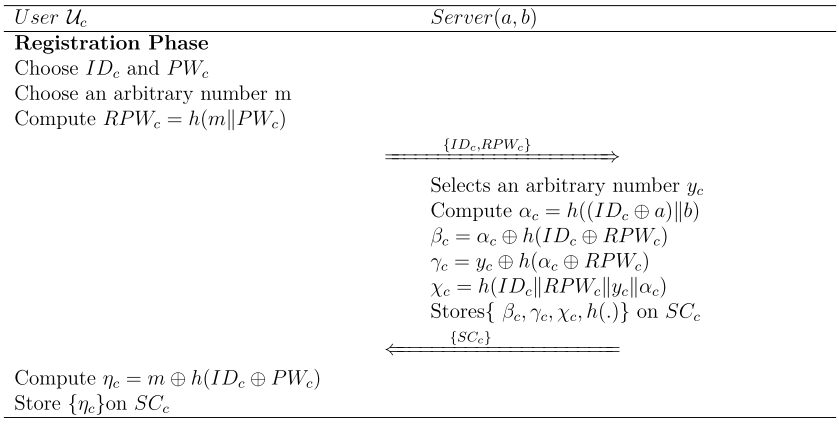

Fig. 1. Registration phase of Kaul and Awasthi's [28] Scheme.

3 After that, server $\mathcal{S}$ stores security parameters $\{\beta_c, \gamma_c, \chi_c, h(.)\}$ in the smart card's SC_c memory and these parameters are sent to U_c through a secure channel.
4 At the end, U_c stores η_c on SC_c, whereas:

$$\eta_c = m \oplus h(ID_c \oplus PW_c) \tag{5}$$

Now, smart card has these parameters $\{\beta_c, \gamma_c, \chi_c, \eta_c, h(\cdot)\}$.

3.2. Login phase

The login phase is shown in Fig. 2. In this section, legitimate U_c transmits a login request by inserting his SC_c into the machine. To generate the login request, the following steps are performed by U_c and card reader:

1. U_c enters his ID_c^* and PW_c^* into the machine.
2. First of all, card reader gets $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$ to calculate:

$$RPW_c^* = h(m \parallel PW_c^*) \tag{6}$$

3. After that, card reader retrieves α_c^* and y_c^* from smart card to calculate χ_c^* as follow:

$$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*) \tag{7}$$

$$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*) \tag{8}$$

$$\chi_c^* = h(ID_c^*\|RPW_c^* \parallel y_c^*\|\alpha_c^*) \tag{9}$$

If the calculated χ_c^* is equivalent to the χ_c that is maintained in smart card/device, then the login request of the requested U_c is accepted by the card reader. There is in build predefined limit set for password guessing attack. If the number of the wrong password guessing attempts exceed the predefined limit, then the smart card gets block automatically. Hence it saves from password guessing attack.

4. After authenticating the legitimacy of requested U_c, card reader calculates:

$$\omega_c = y_c \oplus h(ID_c \oplus \alpha_c) \oplus h(ID_c \oplus \alpha_c \oplus T_1) \tag{10}$$

$$\vartheta_c = h(ID_c\|\alpha_c\|y_c\|(\alpha_c \oplus y_c)\|T_1) \tag{11}$$

Then, login request $\{ID_c, \omega_c, \vartheta_c, T_1\}$ is sent to the $\mathcal{S}$.

3.3. Authentication phase

In this section, login request $\{ID_c, \omega_c, \vartheta_c, T_1\}$ is received to $\mathcal{S}$ at time T_1. U_c and $\mathcal{S}$ verify each other in the following steps:

1. First of all, $\mathcal{S}$ verifies the freshness of the message by checking $(\dot{T} - T_1) \leq \delta T_c$, based on freshness, login request will be accepted or rejected.
2. Server calculates:

$$\alpha_c^* = h((ID_c^* \oplus a) \parallel b) \tag{12}$$

$$y_c^* = \omega_c^* \oplus h(ID_c^* \oplus \alpha_c^*) \oplus h(ID_c^* \oplus \alpha_c^* \oplus T_1) \tag{13}$$

$$\vartheta_c^* = h(ID_c^*\|\alpha_c^*\|y_c^*\|(\alpha_c^* \oplus y_c^*)\|T_1) \tag{14}$$

Then, $\mathcal{S}$ verifies ϑ_c^* with ϑ_c to check the authenticity of login message. If they are not equal, then the session will be aborted immediately.
3. After authentication of ϑ_c, $\mathcal{S}$ further calculates:

$$\mu_c = h(ID_c\|y_c\|(\alpha_c \oplus y_c) \parallel T_2) \tag{15}$$

Where T_2 is contemplated as current time stamp of $\mathcal{S}$. After that, for authentication, $\mathcal{S}$ sends $\{\mu_c, T_2\}$ to card reader .
4. While receiving the challenge message $\{\mu_c, T_2\}$ from $\mathcal{S}$, U_c validates T_2, then calculates:

$$\mu_c^* = h(ID_c\|y_c\|(\alpha_c \oplus y_c) \parallel T_2) \tag{16}$$

Further, U_c verifies μ_c^* with μ_c to check the authenticity of server. If these values are equal, then it means that the server is legitimate.

User $\mathcal{U}_c$	*Server* $\mathcal{S}$
Login and Authentication Phase	
Input ID_c^*, PW_c^*	
Compute $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$	
$RPW_c^* = h(m\|PW_c^*)$	
$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$	
$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*)$	
$\chi*_c = h(ID_c^*\|RPW_c^*\|y_c^*\|\alpha_c^*)$	
Verify $\chi_c^* \stackrel{?}{=} \chi_c$	
Compute $\omega_c = y_c \oplus h(ID_c \oplus \alpha_c) \oplus h(ID_c \oplus \alpha_c \oplus T_1)$	
$\vartheta_c = h(ID_c\|\alpha_c\|y_c\|(\alpha_c \oplus y_c)\|T_1)$	
	$\xrightarrow{\{ID_c,\omega_c,\vartheta_c,T_1\}}$
	Verify $(\acute{T} - T_1) \leq \delta T_c$
	Compute $\alpha_c^* = h((ID_c^* \oplus a)\|b)$
	$y_c^* = \omega_c^* \oplus h(ID_c^* \oplus \alpha_c^*) \oplus h(ID_c^* \oplus \alpha_c^* \oplus T_1)$
	$\vartheta_c^* = h(ID_c^*\|\alpha_c^*\|y_c^*\|(\alpha_c^* \oplus y_c^*)\|T_1)$
	Verify $\vartheta_c^* \stackrel{?}{=} \vartheta_c$
	Compute $\mu_c = h(ID_c\|y_c\|(\alpha_c \oplus y_c)\|T_2)$
	$\xleftarrow{\{\mu_c,T_2\}}$
Verify $(\acute{T_2} - T_2) \leq \delta T_s$	
Compute $\mu_c^* = h(ID_c\|y_c\|(\alpha_c \oplus y_c)\|T_2)$	
Verify $\mu_c^* \stackrel{?}{=} \mu_c$	
Initiate SK	
$SK = h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2)$	
Input ID_c^*, PW_c^*, PW_c^N	
Compute $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$	
$RPW_c^* = h(m\|PW_c^*)$	
$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$	
$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*)$	
$\chi_c^* = h(ID_c^*\|RPW_c^*\|y_c^*\|\alpha_c^*)$	
Verify $\chi_c^* \stackrel{?}{=} \chi_c$	
Compute $RPW_c^N = h(m\|PW_c^N)$	
$\beta_c^N = \alpha_c \oplus h(ID_c \oplus RPW_c)$	
$\gamma_c^N = y_c \oplus h(\alpha_c \oplus RPW_c)$	
$\chi_c^N = h(ID_c\|RPW_c\|y_c\|\alpha_c)$	
$\eta_c^N = m \oplus h(ID_c \oplus PW_c)$	
Update $\beta_c^N, \gamma_c^N, \chi_c^N, \eta_c^N$ on smart card	

Fig. 2. And Authentication phase of Kaul and Awasthi's [28] Scheme.

5. After authentication, the session key can be initiated among both U_c and S for further communication:

$$SK = h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2) \tag{17}$$

3.4. Password change phase

This section is about security. If U_c wishes to change his PW_c, the card reader performs the following steps. Usually, card reader updates the password without connecting to the S:

1. U_c enters his identity ID_c, password PW_c, new password PW_c^N and sends request to card reader to change his previous password.
2. After that, card reader retrieves α_c^* and y_c^* from smart card to calculate χ_c^*:

$$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*) \tag{18}$$

$$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*) \tag{19}$$

$$\chi_c^* = h(ID_c^* \| RPW_c^* \parallel y_c^* \| \alpha_c^*) \tag{20}$$

Then, the card reader verifies the calculated χ_c^* with the χ_c that is kept in the smart card. If this condition does not hold true, then the request is terminated otherwise accepted by the card reader, which computes further to change U_c's password.

3. For registered U_c, card reader calculates:

$$RPW_c^N = h(m \parallel PW_c^N) \tag{21}$$

$$\beta_c^N = \alpha_c \oplus h(ID_c \oplus RPW_c^N) \tag{22}$$

$$\gamma_c^N = y_c \oplus h(\alpha_c \oplus RPW_c^N) \tag{23}$$

$$\chi_c^N = h(ID_c \| RPW_c^N \parallel y_c \| \alpha_c) \tag{24}$$

$$\eta_c^N = m \oplus h(ID_c \oplus PW_c^N) \tag{25}$$

Card reader, then, updates the parameters $\{\beta_c, \gamma_c, \chi_c, \eta_c\}$ by $\{\beta_c^N, \gamma_c^N, \chi_c^N, \eta_c^N\}$ on SC_c.

4. Cryptanalysis of Kaul and Awasthi's scheme

This section performs cryptanalysis of Kaul and Awasthi's scheme that is shown in Fig. 3.

4.1. User impersonation attack

In Kaul and Awasthi's protocol, the server does not keep a record of the values of the user's identities, that are received during various registration requests submitted by various users. Further, the server chooses a unique random number y_c corresponding to ID_c of U_c during registration. But the server does not store the random number y_c corresponding to the registration of an innocent user U_c having identity ID_c. Whenever, the server receives the login request $\{ID_c, \omega_c, \vartheta_c, T_1\}$

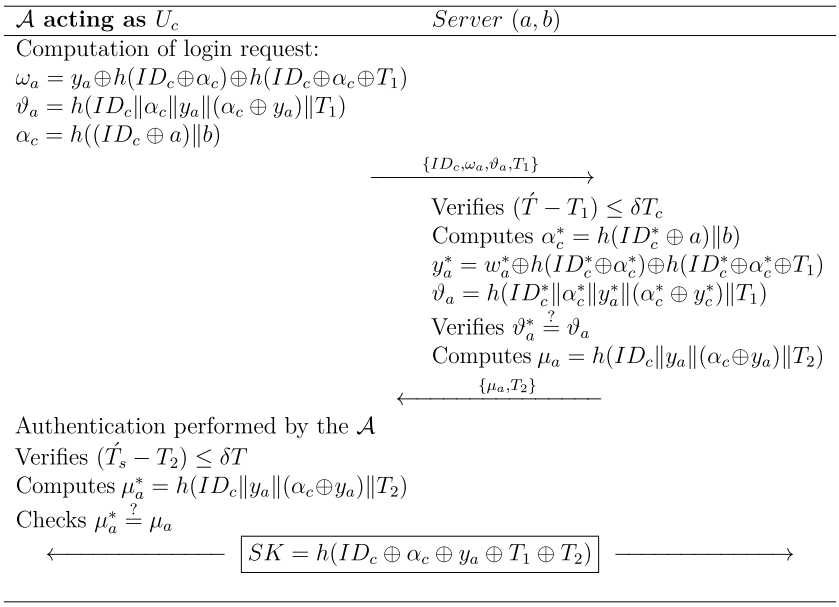

Fig. 3. Impersonation attack on Kaul and Awasthi's [28] Scheme.

from U_c, the server obtains the value of y_c from ω_c. Thus, an attacker $\mathcal{A}$ can take benefit of the facts as mentioned earlier to impersonate as an innocent user U_c as described below:

Step UA 1 $\mathcal{A}$ intercepts the login request $\{ID_c, \omega_c, \vartheta_c, T_1\}$ of an innocent user U_c from open network to get ID_c of U_c. $\mathcal{A}$ chooses a password PW_a and arbitrary number m_a to calculate:

$$RPW_a = h(m_a \parallel PW_a) \tag{26}$$

$\mathcal{A}$ submits $\{ID_c, RPW_a\}$ as registration request to the server.

Step UA 2 Here, it is noticeable that during the registration phase at the server, there is no provision of keeping a track record of the number of times a particular identity has been submitted under previously received registration requests. Therefore, when $\mathcal{A}$ submits the registration request ID_c, RPW_a involving the identity ID_c of U_c and PW_a, m_a chosen by itself, the server performs computation as if it has received the registration request from a new user. Here follows the actions carried by the server on receiving ID_c, RPW_a from $\mathcal{A}$. The server chooses a unique number y_a and computes:

$$\alpha_c = h((ID_c \oplus a) | \ b) \tag{27}$$

$$\beta_a = \alpha_c \oplus h(ID_c \oplus RPW_a) \tag{28}$$

$$\gamma_a = y_a \oplus h(\alpha_c \oplus RPW_a) \tag{29}$$

$$\chi_a = h(ID_c | \ RPW_a | \ y_a | \ \alpha_c) \tag{30}$$

Server stores $\{\beta_a, \gamma_a, \chi_a, h(.)\}$ on smart card and forwards it to $\mathcal{A}$ (who is acting as a user willing to get registered at the server) securely.

Step UA 3 On receiving the smart card, $\mathcal{A}$ extracts the values of $\{\beta_a, \gamma_a, \chi_a, h(.)\}$. Then, $\mathcal{A}$ retrieves α_c and y_a by computing:

$$\alpha_c = \beta_a \oplus h(ID_c \oplus RPW_a) \tag{31}$$

$$y_a = h(\alpha_c \oplus RPW_a) \oplus \gamma_a \tag{32}$$

where $RPW_a = h(m_a | \ PW_a)$. Now, having α_c and y_a, $\mathcal{A}$ can impersonate as the innocent user U_c as shown below:

5. Proposed scheme

To remove security issues in the protocol of Kaul and Awasthi [28], in this section, we have illustrated an enhanced distant user authentication protocol with SK agreement, which keeps all the basic characteristics of Kaul and Awasthi's [28] scheme. Moreover, our scheme resolves all the security problems to make the protocol secure and effective for real-world applications. Similar to the Kaul and Awasthi's [28] scheme, our presented scheme consists of four main phases: the registration phase, the login phase, the authentication phase, and the password change phase.

5.1. Registration phase

The registration phase of proposed protocol is given in Fig. 4. As per the proposed protocol, user U_c and S perform below-mentioned steps to register the cth user on to the remote server.

1. Firstly, the cth user U_c selects his identity ID_c, password PW_c and an arbitrary number m. After that, U_c calculates $RPW_c = h(m \parallel PW_c)$, and sends request $\{ID_c, RPW_c\}$ to S via secure channel.
2. Then, S selects an arbitrary number y_c for U_c to calculate the following values:

$$\overline{DID_c} = Enc_{ds}(ID_c \parallel y_c) \tag{33}$$

$$\alpha_c = h((ID_c \oplus a) \parallel b) \tag{34}$$

$$\beta_c = \alpha_c \oplus h(ID_c \oplus RPW_c) \tag{35}$$

$$\gamma_c = y_c \oplus h(\alpha_c \oplus RPW_c) \tag{36}$$

$$\chi_c = h(ID_c \| RPW_c \parallel y_c \| \alpha_c) \tag{37}$$

3. After these calculations, S stores the calculated values in smart card $\{\beta_c, \gamma_c, \overline{DID_c}, \chi_c, h(.)\}$ and sends towards U_c via protected channel.
4. At the end, U_c inserts η_c in smart card as:

$$\eta_c = m \oplus h(ID_c \oplus PW_c) \tag{38}$$

Now, smart card has these parameters $\{\beta_c, \gamma_c, \overline{DID_c}, \chi_c, h(.), \eta_c\}$.

$User\ \mathcal{U}_c$	$Server\ \mathcal{S}$
Choose ID_c and PW_c	
Choose an arbitrary number m	
Compute $RPW_c = h(m\|PW_c)$	
	$\xrightarrow{\{ID_c, RPW_c\}}$
	Selects an arbitrary number y_c
	$\boxed{\overline{DID_c} = Enc_{ds}(ID_c\|y_c)}$
	Compute $\alpha_c = h((ID_c \oplus a)\|b)$
	$\beta_c = \alpha_c \oplus h(ID_c \oplus RPW_c)$
	$\gamma_c = y_c \oplus h(\alpha_c \oplus RPW_c)$
	$\chi_c = h(ID_c\|RPW_c\|y_c\|\alpha_c)$
	Stores$\{\ \beta_c, \gamma_c, \overline{DID_c}, \chi_c, h(.)\}$ on SC_c
	$\xleftarrow{\{SC_c\}}$
Compute $\eta_c = m \oplus h(ID_c \oplus PW_c)$	
Store $\{\eta_c\}$on SC_c	

Fig. 4. Registration phase of proposed scheme.

5.2. Login phase

When a registered user U_c wishes to login into the system, he enters his/her smart card into the SC_c reader that performs the following calculations:

1. U_c inputs his ID_c^* and PW_c^* in smart card reader.
2. Smart card extracts $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$ and calculates $RPW_c^* = h(m \parallel PW_c^*)$.
3. Further, smart card reader derives $\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$ and $y_c^* = \gamma_c \oplus h(\alpha^* \oplus RPW_c^*)$ for calculating:

$$\chi_c^* = h(ID_c^*\|RPW_c^* \parallel y_c^*\|\chi_c^*) \tag{39}$$

If $\chi_c^* = \chi_c$ then, it will accept login request of U_c; otherwise, a request would be rejected.

4. After validating the authenticity of the legitimate U_c, card reader calculates:

$$\omega_c = y_c \oplus (ID_c \oplus \alpha_c) \oplus h(ID_c \oplus \alpha_c \oplus T_1) \tag{40}$$

$$\vartheta_c = h(ID_c\|\alpha_c\|y_c\|(\alpha_c \oplus y_c)\|T_1) \tag{41}$$

Card reader, then, sends login request $\{ID, \omega_c, \vartheta_c, T_1\}$ to $\mathcal{S}$.

5.3. Authentication phase

The login and authentication phase is depicted in Fig. 5. In this phase, server $\mathcal{S}$ receives login request $\{ID, \omega_c, \vartheta_c, T_1\}$ on time T_1. After that, smart card reader and $\mathcal{S}$ performs under-mentioned calculations to authenticate one other:

1. $\mathcal{S}$, at first, checks the legitimacy of the time stamp T_1 by verifying $(\acute{T} - T_1) \leq \delta T$, if the value is less than the defined threshold, then $\mathcal{S}$ accepts login request otherwise rejects it.
2. After that, $\mathcal{S}$ extracts ID_c as $(ID_c \parallel y_c) = Dec_{ds}(\overline{DID_c})$ and calculates following values:

$$\alpha_c^* = h((ID_c^* \oplus a) \parallel b) \tag{42}$$

$$y_c^* = \omega_c^* \oplus h(ID_c^* \oplus \alpha_c^*) \oplus h(ID_c^* \oplus \alpha_c^* \oplus T_1) \tag{43}$$

$$\vartheta_c^* = h(ID_c^*\|\alpha_c^*\|y_c^*\|(\alpha_c^* \oplus y_c^*)\|T_1) \tag{44}$$

$\mathcal{S}$, then, validates the legitimacy of login request by making a comparison of calculated ϑ_c^* with the stored ϑ_c and if it is not validated, then $\mathcal{S}$ rejects this request otherwise accepts it.

3. After the validation of ϑ_c, $\mathcal{S}$ calculates:

$$\mu_c = h(ID_c\|y_c \parallel (\alpha_c \oplus y_c)\|T_2) \tag{45}$$

Here, T_2 is the present time, $\mathcal{S}$ sends $\{\mu_c, T_2\}$ to the U_c for authentication.

4. While receiving the challenge message $\{\mu_c, T_2\}$ from $\mathcal{S}$, U_c validates T_2 and calculates:

$$\mu_c^* = h(ID_c\|y_c\|(\alpha_c \oplus y_c) \parallel T_2) \tag{46}$$

U_c validates μ_c^* with μ_c for authenticating the $\mathcal{S}$. If both μ_c and μ_c^* are equal, then the $\mathcal{S}$ is authenticated.

5. After authenticating each other, SK is calculated:

$$SK = h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2) \tag{47}$$

5.4. Password change phase

If U_c is willing to change his/her password by replacing PW_c with PW_c^N, then calculations that are performed by SC to update the password without communicating with the $\mathcal{S}$ are given below:

1. To change his password, U_c enters his ID_c, PW_c and PW_c^N.
2. Smart card reader extracts $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$ to calculate $RPW_c^* = h(m \parallel PW_c^*)$
3. After that, smart card reader extracts $\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$ and $y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*)$ to calculate:

$$\chi_c^* = h(ID_c^*\|RPW_c^* \parallel y_c^*\|\alpha_c^*) \tag{48}$$

If the already stored χ_c in the smart card is equal to calculated χ_c^*, then the smart card accepts the request for changing the password; otherwise, it rejects the request for changing password.

4. For a legal U_c, smart card reader calculates following values:

$$RPW_c^N = h(m \parallel PW_c^N) \tag{49}$$

$$\beta_c^N = \alpha_c \oplus h(ID_c \oplus RPW_c) \tag{50}$$

$$\gamma_c^N = y_c \oplus h(\alpha_c \oplus RPW_c) \tag{51}$$

$$\chi_c^N = h(ID_c\|RPW_c \parallel y_c\|\alpha_c) \tag{52}$$

$$\eta_c^N = m \oplus h(ID_c \oplus PW_c) \tag{53}$$

Update $\{\beta_c, \gamma_c, \chi_c, \eta_c\}$ with $\{\beta_c^N, \gamma_c^N, \chi_c^N, \eta_c^N\}$ on smart card.

6. Security analysis

This section consists of the introduced scheme's security analysis. Both formal and informal analysis are discussed in detail.

6.1. Informal security analysis

This section extensively describes informal security analysis and proves that our proposed scheme is safe against the following attacks:

$User\ \mathcal{U}_c$	$Server\ \mathcal{S}$
Login and Authentication Phase	
Input ID_c^*, PW_c^*	
Compute $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$	
$RPW_c^* = h(m \| PW_c^*)$	
$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$	
$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*)$	
$\chi_c^* = h(ID_c^* \| RPW_c^* \| y_c^* \| \alpha_c^*)$	
Verify $\chi_c^* \stackrel{?}{=} \chi_c$	
Compute $\omega_c = y_c \oplus h(ID_c \oplus \alpha_c) \oplus h(ID_c \oplus$	
$\alpha_c \oplus T_1)$	
$\vartheta_c = h(ID_c \| \alpha_c \| y_c \| (\alpha_c \oplus y_c) \| T_1)$	
	$\xrightarrow{\{\overline{DID_c}, \omega_c, \vartheta_c, T_1\}}$
	Verify $(\acute{T} - T_1) \leq \delta T_c$
	$\boxed{(ID_c \| y_c) = Dec_{ds}(\overline{DID_c})}$
	Compute $\alpha_c^* = h((ID_c^* \oplus a) \| b)$
	$y_c^* = \omega_c^* \oplus h(ID_c^* \oplus \alpha_c^*) \oplus h(ID_c^* \oplus \alpha_c^* \oplus T_1)$
	$\vartheta_c^* = h(ID_c^* \| \alpha_c^* \| y_c^* \| (\alpha_c^* \oplus y_c^*) \| T_1)$
	Verify $\vartheta_c^* \stackrel{?}{=} \vartheta_c$
	Compute $\mu_c = h(ID_c \| y_c \| (\alpha_c \oplus y_c) \| T_2)$
	Generates a new random number
	Computes $\boxed{PID_c = Enc_{ds}(ID_c \| y_{new}) \oplus a_c^*}$
	$\xleftarrow{\{\mu_c, T_2\}}$
Verify $(\acute{T}_2 - T_2) \leq \delta T_s$	
Compute $\mu_c^* = h(ID_c \| y_c \| (\alpha_c \oplus y_c) \| T_2)$	
Verify $\mu_c^* \stackrel{?}{=} \mu_c$	
$\boxed{\overline{DID_c} = PID_c \oplus a_c^*}$	
Initiate SK	
$SK = h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2)$	
Input ID_c^*, PW_c^*, PW_c^N	
Compute $m = \eta_c \oplus h(ID_c^* \oplus PW_c^*)$	
$RPW_c^* = h(m \| PW_c^*)$	
$\alpha_c^* = \beta_c \oplus h(ID_c^* \oplus RPW_c^*)$	
$y_c^* = \gamma_c \oplus h(\alpha_c^* \oplus RPW_c^*)$	
$\chi_c^* = h(ID_c^* \| RPW_c^* \| y_c^* \| \alpha_c^*)$	
Verify $\chi_c^* \stackrel{?}{=} \chi_c$	
Compute $RPW_c^N = h(m \| PW_c^N)$	
$\beta_c^N = \alpha_c \oplus h(ID_c \oplus RPW_c)$	
$\gamma_c^N = y_c \oplus h(\alpha_c \oplus RPW_c)$	
$\chi_c^N = h(ID_c \| RPW_c \| y_c \| \alpha_c)$	
$\eta_c^N = m \oplus h(ID_c \oplus PW_c)$	
Update $\beta_c^N, \gamma_c^N, \chi_c^N, \eta_c^N$ on smart card	

Fig. 5. Login and authentication phase of proposed scheme.

6.1.1. Privileged insider attack

In our enhanced scheme, U_c's password is not sent in a simple text rather than the password is hashed as $RPW_c = h(m \parallel PW_c)$; then it is sent via a secure channel. Consequently, it is not possible for an insider to find the values of parameters m and PW_c in a specific time that will resist an $\mathcal{A}$ to use private information of U_c for his own gain. So, the presented scheme is secure against insider attack.

6.1.2. Smart card stolen attack

If U_c lost his/her smart card, then an attacker can easily extract the values stored in smart card $\{\beta_c, \gamma_c, \overline{ID_c}, \chi_c, h(.)\}$ through power analysis. After getting these values, still computationally it is not feasible for an attacker to find the arbitrary number y_c, password PW_c, secret key a and server's secret number b. Therefore, chances to correctly guess these values are near to impossible in a feasible time. Hence, the proposed scheme prevents smart card stolen attack.

6.1.3. User and server impersonation attack

For the prevention of both user and the server impersonation attacks, the values ω_c, ϑ_c and μ_c are made secure with hash functions. If an adversary wishes to impersonate himself as legal U_c, he will send login request $\{\overline{ID_c}, \omega_c, \vartheta_c, T_1\}$ to the server but will not pass the validity assessment $\vartheta_c \stackrel{?}{=} \vartheta_c$. Moreover, the server impersonation will be detected by the user by validating μ_c against forged authentication request $\{\mu_c, T_2\}$. If the attacker wants to change ϑ_c and μ_c, then he needs to correctly calculate these values m, y_c, a, b and PW_c that are near to impossible in limited time duration. Hence, our presented scheme resists against impersonation attacks.

6.1.4. Online password guessing attack

For $\mathcal{A}$, to guess the password of legitimate user online, he inputs the guessed ID_c and PW_c of the legal user. Card reader checks either inputted ID_c and PW_c are correct or not by verifying χ_c, before the calculation of any request message. The smart card is blocked by the card reader if the wrong ID_c and PW_c are inputted more than the defined limit of the card reader. Therefore, it becomes impossible for an $\mathcal{A}$ to perform this attack.

6.1.5. Offline password guessing attack

An attacker can eavesdrop the request message $\{ID_c, \omega_c, \vartheta_c, T_1\}$, authentication message $\{\mu_c, T_2\}$ and the important credentials which are maintained in user's smart card $\{\beta_c, \gamma_c, \chi_c, h(.)\}$. After that, from his

own directory, adversary will try to guess the parameters y_c, a, b, PW_c offline. At mean time, at least two unknown secret parameters must be correctly guessed by $\mathcal{A}$, that is not possible. So, due to this property, our presented protocol is secure against this threat.

6.1.6. Replay attack

Replay of the login and the authentication messages $\{ID_c, \omega_c, \vartheta_c, T_1\}$ and $\{\mu_c, T_2\}$ is not useful for an adversary as there involves time stamp in each transmitted message. Further, by verifying freshness of transmitted messages ϑ_c, μ_c, and time stamps, authenticity is being checked. So, an adversary cannot replay to any message.

6.1.7. Denial of services attack

This attack is possible if an attacker sends a fake request message many times to $\mathcal{S}$. If a card reader receives the value of failure login request higher than the system's predefined limit, then at that time, the card reader blocks the card for the time being as it saves energy, computation resources and time of $\mathcal{S}$. Therefore, the introduced scheme is proved to be secure against denial of service attacks.

6.1.8. Man-in-middle-attack

An adversary can manipulate, intercept, and eavesdrop the information that is transmitted between U_c and $\mathcal{S}$; it is said to be man-in-middle-attack. If an attacker captures these messages even then, he/she cannot take advantage of it, because all of these parameters $\overline{DID_c}, \mu_c, \omega_c$ are secured with hashing function, encryption, and decryption. Therefore, it is not possible to find the values of these parameters in a feasible time. For this reason, our introduced scheme resists this attack.

6.1.9. Mutual authentication

In presented scheme, the $\mathcal{S}$ verifies the authenticity of U_c by ϑ_c as it includes U_c's identity ID_c and password PW_c. Moreover, U_c verifies $\mathcal{S}$ by checking $\mu_c^* \stackrel{?}{=} \mu_c$, where $\mu_c = h(ID_c \| y_c \| (\alpha_c \oplus y_c) \parallel T_2)$ and to calculate μ_c, server's secret key is required. So, introduced scheme offers mutual authentication, as only legal server $\mathcal{S}$ and legal U_c can authenticate each other.

6.2. Formal security analysis

In this section, we will talk about the random oracle model, that is utilized to prove enhanced protocol's security.

6.2.1. Security model

To verify our enhanced protocol against several attacks, we are going to use the security model. The selected model is described below:

Participants

The number of communicants in a network are executed in authentication scheme . In a network, every communicant can be a server $\mathcal{S} \in \mathcal{S}$ or user $U \in \mathcal{U}$. Perhaps, it is possible that the various entities of each communicant act as oracle and each of oracle is absolutely associated with the unique execution of . Affiliating to $\mathcal{U}^s$ nth occurrence $(rep.S)$ in unique session as ${}^n_{\mathcal{U}}$ (rep. ${}^s_{\mathcal{U}}$). ${}^n_{\mathcal{U}}$ (rep. ${}^s_{\mathcal{U}}$) is linked with ID and $\overline{DID^k_{\mathcal{U}}}$ (rep. $\overline{DID^s_{\mathcal{U}}}$) with session $IDsid^n_{\mathcal{U}}$ (rep. $sid^s_{\mathcal{U}}$) as well as session key $SK^k_{\mathcal{U}}$. $\overline{DID^n_{\mathcal{U}}}$ (rep. $\overline{DID^s_{\mathcal{U}}}$) where $\overline{DID^n_{\mathcal{U}}}$ (rep. $\overline{DID^s_{\mathcal{U}}}$) shows the set of engaged identities in suggested entities while sid^s_U (rep. sid^J_S) display the flow that have been sent and received by ${}^n_{\mathcal{U}}$ (rep. ${}^K_S$). ${}^n_{\mathcal{U}}$ (rep. ${}^K_S$) is supposed to be approved, if it grasp the session key $SK^n_{\mathcal{U}}$ (rep. SK^k_S). All the identifiers $\overline{DID^n_{\mathcal{U}}}$ (rep. $\overline{DID^s_{\mathcal{U}}}$) , $sid^s_{\mathcal{U}}$ (rep. sid^n_S) , ${}^k_{\mathcal{U}}$ and ${}^k_S$ are supposed to be true participants if (1) both are approved (2) $sid^k_S = sid^n_{\mathcal{U}}$ (3) ${}^n_{\mathcal{U}} = {}^k_S$ (4) $\overline{DID^n_{\mathcal{U}}} = \overline{DID^k_S}$.

Long lived key

Each $U \in \mathcal{U}$ holds a unique $PW_{\mathcal{U}}$ and each $S \in \mathcal{S}$ carries the vector $PW_{\mathcal{S}}$ with each associated entry to every user.

Adversary model

Let us suppose that an attacker $\mathcal{A}$ can control the communication channel. Adversary $\mathcal{A}$ fetches personal credentials and then proceed the session between server and user. After that, $\mathcal{A}$ could execute the following steps in any sequence.

- *Execute*$({}^n_{\mathcal{U}}, {}^k_S)$ This query is utilized for making passive attacks possible for adversary $\mathcal{A}$. In order to entrap, this query can be run by $\mathcal{A}$ on the execution between ${}^n_{\mathcal{U}}$ and ${}^k_S$. This query shows exchanged messages between server and user.
- *SendClient*$({}^n_{\mathcal{U}}, msg)$ This query is used to make active attacks possible for adversary $\mathcal{A}$. It means that $\mathcal{A}$ can easily change, intercept and create a new message or send this message to ${}^n_{\mathcal{U}}$. On receiving the message msg, the message created by ${}^n_{\mathcal{U}}$ can be displayed by using his query.
- *SendServer*$({}^k_S, msg)$ This query is used to make an adversary $\mathcal{A}$ able to run an active attack across $S \in \mathcal{S}$. On receiving the message msg, $\mathcal{A}$ uses this query to intercept the message created by ${}^k_S$.
- *Reveal* $({}^n_{\mathcal{U}})$ This query is used to obtain the session key SK of ${}^n_{\mathcal{U}}$.
- *Corrupt* $(\mathcal{U})$ Long lived key of user $\mathcal{U}$ can be displayed by using this query.
- *Test* $({}^n_{\mathcal{U}})$ Adversary $\mathcal{A}$ manipulates any such query to fresh the oracle. An arbitrary bit i.e., $B \in \{0, 1\}$ is generated in the response of this query, if $B = 0$ an arbitrary value is returned, otherwise the session key SK of ${}^n_{\mathcal{U}}$ is returned back.

Fresh oracle An oracle ${}^n_{\mathcal{U}}$ is fresh, if it can be declared in the case that (1) ${}^n_{\mathcal{U}}$ has accepted for approval (2) Once reveal query has been accepted then it does not crack either by ${}^n_{\mathcal{U}}$ or any of its participant.

Protocol security The security of can be displayed easily by using $GAME(, \mathcal{A})$. An $\mathcal{A}$ can execute a number of queries that are already defined to ${}^n_{\mathcal{U}}$ and ${}^k_S$ during the simulation period of this game. If an attacker $\mathcal{A}$ declares that a query Test $({}^n_{\mathcal{U}})$ and $({}^k_S)$ have approved and its *fresh* as well then $\mathcal{A}$ displays a bit. $\mathcal{A}$ guesses B successfully. The asset of the $\mathcal{A}$ is specified below:

$$Adt_{,UDD}(\mathcal{A}) = |4Pr[B = B'] - 4| \tag{54}$$

If $Adt_{,UDD}(\mathcal{A})$ is negligible, then is supposed to be secure.

6.2.2. Security proof

Theorem 1. *UDD stands for Uniformly-distributed dictionary of exclusively feasible passwords that have size $|UDD|$, and the enhanced protocol is explained by . Assume that the non-collisional hashing function is modeled as the random oracle. Later on,*

$$Adt_{,UDD}(\mathcal{A}) \leq \frac{q^2_{Hq} + (q_{Send} + q_{Exe})^2}{2^{len}} + \frac{q_{Hq}}{2^{len}} + \frac{q_{Send}}{|UDD|} \tag{55}$$

where, q_{Send} displays entire $Send$ queries, q_{Exe} displays entire $Execute$ queries and q_{Hq} displays the entire number of hash function queries.

Proof. This proof consists of a game fusion that began from UA 0 and ended at UA 3. While $\mathcal{A}$ has no benefit, for every $UA_x (0 \leq x \leq 3)$. $Succ_z$ is described as an isolated event that $\mathcal{A}$ tries to guess B successfully for an isolated test session.

GAME UA 0

In this game section, entire $S \in \mathcal{S}$ and $U \in \mathcal{U}$ are run in random oracle. According to the definition, as mentioned above of event $Succ_z$ by using $Test$ Query, an $\mathcal{A}$ can guess B accurately, we got:

$$Adt_{,UDD}(\mathcal{A}) = 2|Pr[Succ_0] - 0.5| \tag{56}$$

GAME UA 1

This game is same as game UA 0 but the random oracle r creates a hash list h_{list} whereas, all the rows in h_{list} are in the form (RP, EP). UA 1 outputs RP, if a row (RP, EP) presents in h_{list} otherwise, randomly

selected $RP \in \{0,1\}$ is sent to Adv and kept new row (RP, EP) in h_{list}. Entire server and user instances are run for *Send*, *Execute*, *SendClient*, *SendServer*, *Reveal*, *Corrupt* and *Test* queries. Absolutely, it can be easily justifiable that this game is secure across all known attacks.

$$Pr[Succ_0] = Pr[Succ_1] \tag{57}$$

GAME UA 2

All the executions that are discussed in UA 1 are included in this game. In addition, if the collision is appeared in this game between small transcripts $S_k \Delta a$ and hash H values, then this game is rejected. Suggesting the paradox's birthday, $(q_{Send}+q_{Exe})^2/2^{len+1}$ is the maximum chances of collision in the result of a transcript, where Hq is the possible maximal number of the hashed queries. Analogously, the maximal occurrences of collision is $q_{Hq}^2/2^{len+1}$ in the output of the hashed oracles, where the maximum available number of the queries that is $Send$ to the oracle are q_{Send} and q_{Exe}, and len describes the length of the bits of numbers that are randomly generated, it also indicates the hashed functions output, we obtain:

$$|Pr[Succ_2] - Pr[Succ_1]| \leq \frac{q_{Hq}^2 + (q_{Send} + q_{Exe})^2}{2^{len+1}} \tag{58}$$

GAME UA 3

In this phase, execution of all possible queries to $SendClient$ oracle have now modified for the sessions that are chosen in UA 2. To make the session key SK independent from password and all other related keys, its calculation is changed. We $Send$ $(^n_{\mathcal{U}}, \mu_c, T_2)$ as well as $Send$ $(^k_S, \overline{DID_c}, \omega_c, \vartheta_c, T_1)$ are inquired. We calculate $SK = h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2)$, where y_c is chosen randomly. There are two possible cases that are given below where UA 2 and UA 3 are entirely different:

- **CASE UE 1**: $\mathcal{A}$ queries $h(ID_c \oplus \alpha_c \oplus y_c \oplus T_1 \oplus T_2)$ to H. The chances of appearance of event that is above said are $q_{Hq}/2^{len}$.
- **CASE UE 2**: $\mathcal{A}$ responds $Send$ query without $Send$ $(^n_{\mathcal{U}}, \mu_c, T_2)$ and perfectly cheats the user $\mathcal{U}$. Adversary $\mathcal{A}$ is not able to reveal the private parameters PW_c of user. There is $1/|UDD|$ probability that $\mathcal{A}$ can get user's password, it means that probability of $q_{Send}/|UDD|$ is greater than the appearance probability of said event.

The variation between UA 2 and UA 3 is as follow:

$$|Pr[Succ_3] - Pr[Succ_2]| \leq \frac{q_{hs}}{2^{len}} + \frac{q_{Send}}{|UDD|} \tag{59}$$

and

$$Pr[Succ_3] = 0.5 \tag{60}$$

Following is the result by combining all above equations:

$$\begin{aligned} Adt_{,UDD}(\mathcal{A}) &= 2|Pr[Succ0] - 0.5| \\ &= 2|Pr[Succ_0] - Pr[Succ_3]| \\ \leq 2(|Pr[Succ_1] - Pr[Succ_2] + Pr[Succ_2] - Pr[Succ_3]|) \\ &\leq \frac{q_{Hq}^2 + (q_{Send} + q_{Exe})^2}{2^{len}} + \frac{q_{Hq}}{2^{len}} + \frac{q_{Send}}{|UDD|} \end{aligned} \tag{61}$$

7. Performance and security comparisons

This portion illustrates the complete security and performance analysis of the enhanced protocol, also provides a comparison with other related protocols. To validate the performance of the presented protocol, the inbuilt PyCrypto library is used to implement the cryptographic operations (that are used in our proposed scheme) in Ubuntu 19.04, using a python programming language with system specifications as mentioned in Table 2. To obtain average time, the enhanced protocol is executed various times under the same conditions and using the same tools. The result shows that time which is required for non-collisional hashing function and concatenation is 0.00089 and 0.00014, respectively. In addition, XOR operation and encryption/decryption have a diminutive amount of time. Therefore, these operations are not taken into account. The number of the bits that are required for username, identity, password, XOR operation, P (elliptic curve point), arbitrary number, and integer are 160 bits. Moreover, a non-collisional hashing function, server private key, and server public key take 256 bits. Similarly, encryption/decryption has required 512 bits. Symbols used for cryptographic operations are described below:

T_h time needed for computing hash function.
T_m time needed for computing point multiplication.
$T_{\|}$ time needed for computing concatenation.
$T_{\oplus}$ time required for computing XOR operation.
$T_{Enc/Dec}$ time required for computing encryption/decryption.

Table 2
System specifications.

Item	Specification
Processor	i7 3.60 GHz
RAM	16.0 GB

7.1. Storage cost

This section shows the storage cost of the presented protocol with affiliated protocols. Storage cost is the cost of parameters that are kept in the smart card and database. In our protocol, parameters $\{\beta_c, \gamma_c, \overline{DID_c}, \chi_c, h(.), \eta_c\}$ stored in smart card take 1536 bits. Moreover, the storage costs of the protocols of Kaul and Awasthi [28], Kumari et al. [17] and Chang et al. [16] are 1280 bits, 1696 bits and 672 bits, respectively. Fig. 6 shows that the storage cost of our protocol is less than Kumari et al.'s [17] scheme and slightly greater than Chang et al. [16] and Kaul and Awasthi [28]. In Fig. 6, the number of bits is represented vertically, and protocols are represented horizontally.

7.2. Communication cost

This section illustrates the communication cost of the introduced protocol with affiliated protocols. Communication cost of our protocol is 3296 bits, similarly, communication cost of the protocols of Kaul and Awasthi [28], Kumari et al. [17] and Chang et al. [16] are 2668 bits, 3296 bits and 2336 bits, respectively. Fig. 7 shows that the communication cost of our protocol is equal to the Kumari et al.'s [17] protocol and slightly greater than Chang et al. [16] and Kaul and Awasthi [28].

7.3. Computation cost

Computation cost is the cost of cryptographic operations that are used in our proposed protocol with affiliated protocols. Computation cost is calculated in milliseconds(ms). Computation cost for one hash operation is 0.00089 ms, and for concatenation is 0.00014 ms. Therefore, the computation cost of our protocol is 0.0215 ms bit. Similarly, computation cost of Kaul and Awasthi [28], Kumari et al. [17] and Chang et al. [16] are 0.021 ms, 0.02191 ms and 0.01318 ms, respectively. Fig. 8 shows that the computation cost of the presented protocol is equal to the Kumari et al. [17] and Kaul and Awasthi [28] and slightly greater than Chang et al.'s [16] scheme.

Table 3 describes that the communication cost of enhanced protocol is equivalent to the [17]. Similarly, storage cost of our protocol is less than [17] and slightly greater than [28] and [16]. Moreover, computation cost is equal to the [17,28] and slightly greater than [16]. In addition, our presented scheme is secured against numerous attacks as shown in Table 4.

The introduced scheme is secure against major security attacks such as impersonation, insider, replay, and password guessing attacks. Moreover, the proposed scheme provides mutual authentication and user anonymity. Table 4 presents the comparison of the security features of the proposed scheme with contemporary schemes.

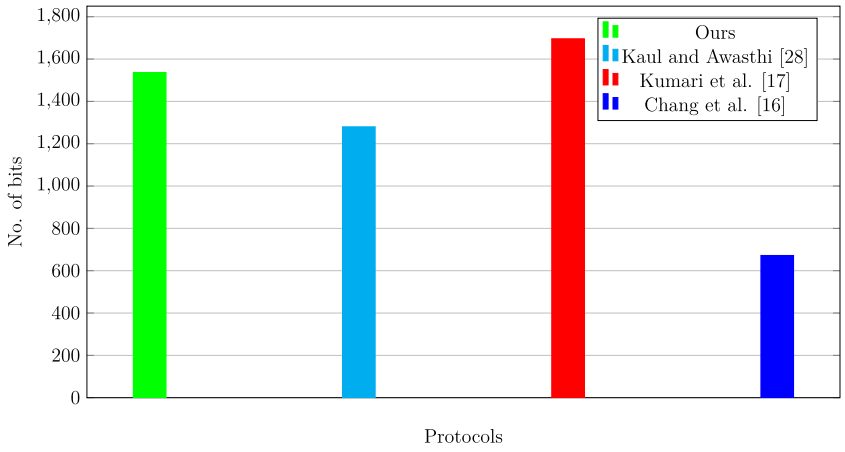

Fig. 6. Analysis of storage cost between proposed and related protocols.

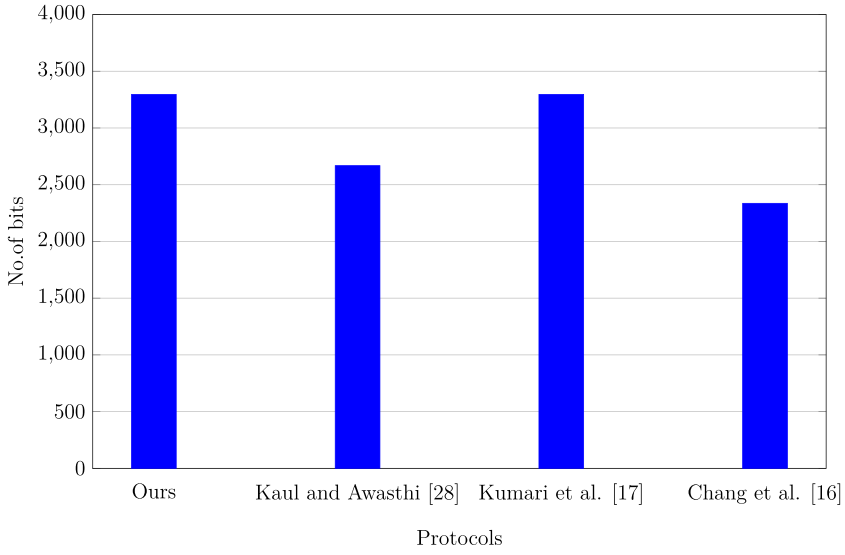

Fig. 7. Analysis of communication cost between proposed and related protocols.

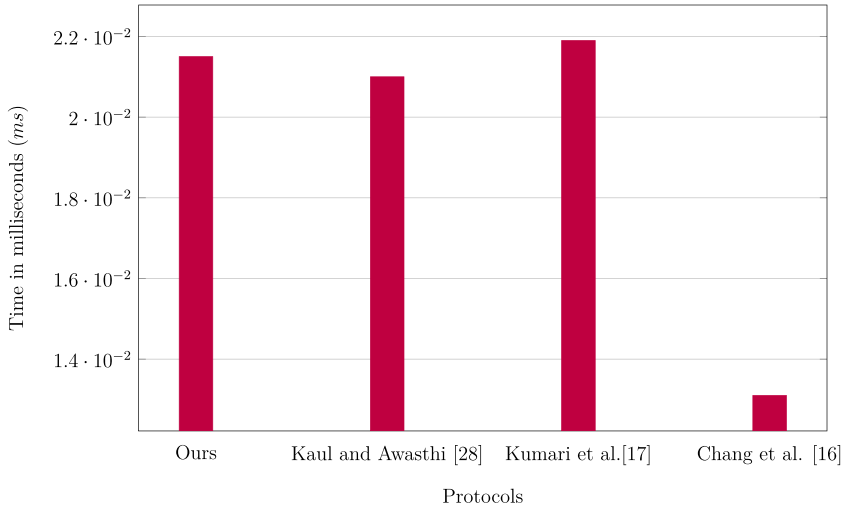

Fig. 8. Analysis of computation cost between proposed and related protocols.

Consequently, by thoroughly analyzing Tables 3 and 4, it can easily be stated that our enhanced scheme takes reasonable bits for storage and communication process, and it needs less time for the computation process. Although our scheme exhibits a little bit more storage and computation cost as compared to related protocols, however, it offers

Table 3
Proposed protocol comparison with affiliated protocols.

	Comp. Cost	Storage Cost	Bits Exch.
Our	$20T_h + 29T_{\oplus} + 27T_{\|\|} + 3T_{Enc/Dec} = 0.0215$ ms	1536 bits	3296 bits
[28]	$20T_h + 28T_{\oplus} + 23T_{\|\|} = 0.021$ ms	1280 bits	2668 bits
[17]	$19T_h + 18T_{\oplus} + 36T_{\|\|} = 0.0219$ ms	1696 bits	3296 bits
[16]	$12T_h + 7T_{\oplus} + 18T_{\|\|} = 0.0131$ ms	672 bits	2336 bits

Table 4
Comparison of security parameters.

Scheme:	Our	[28]	[17]	[16]
Insider Attack	Yes	Yes	No	No
Smart card Stolen Attack	Yes	No	No	Yes
Impersonation Attack	Yes	No	Yes	No
Online Password Guessing Attack	Yes	Yes	Yes	No
Offline Password Guessing Attack	Yes	Yes	Yes	Yes
Replay attack	Yes	Yes	No	No
Denial of Services Attack	Yes	Yes	Yes	Yes
Man in the Middle Attack	Yes	Yes	Yes	Yes
Mutual Authentication	Yes	Yes	No	Yes

enhanced security features that do not exist in [16,17]. Hence, our scheme is more suitable and practical due to aided security features.

8. Conclusion

In this article, we have crypt-analyzed a distant user authenticated key agreement protocol by Kaul and Awasthi and demonstrated that their introduced protocol is not secure for real-life applications. An attacker can masquerade himself as a legal user by easily getting the identity of the legal user, which is being sent on the public channel in plaintext and can take benefit of services provided by the server on behalf of the legal user (victim). So, their claim of being secure is not valid as their presented scheme is vulnerable to user masquerading attacks. Therefore, we have presented an improved protocol in this article to make it secure against the user impersonation attack. Moreover, through formal and informal security analysis, we have demonstrated that our proposed protocol is safe against major security threats while utilizing limited communication, computation, and storage resources. Due to better security and performance proposed protocol is a good candidate for deployment in 6G/IoT infrastructure.

CRediT authorship contribution statement

Minahil Rana: Writing - original draft, Conceptualization, Investigation, Methodology, Formal analysis. **Akasha Shafiq:** Writing - original draft, Investigation, Methodology, Formal analysis. **Izwa Altaf:** Writing - original draft, Investigation, Methodology, Formal analysis. **Mamoun Alazab:** Conceptualization, Writing -review & editing, Investigation, Methodology, Formal analysis, Resources. **Khalid Mahmood:** Writing - original draft, Conceptualization, Writing -review & editing, Investigation, Methodology, Formal analysis, Supervision, Resources. **Shehzad Ashraf Chaudhry:** Writing - original draft, Conceptualization, Writing -review & editing, Investigation, Methodology, Formal analysis, Supervision, Resources. **Yousaf Bin Zikria:** Conceptualization, Writing -review & editing, Investigation, Methodology, Formal analysis, Supervision, Resources.

Declaration of competing interest

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

References

[1] L. Lamport, Password authentication with insecure communication, Commun. ACM 24 (11) (1981) 770–772.
[2] C.-K. Chan, L.-M. Cheng, Cryptanalysis of a remote user authentication scheme using smart cards, IEEE Trans. Consum. Electron. 46 (4) (2000) 992–993.
[3] C.-C. Chang, T.-C. Wu, Remote password authentication with smart cards, IEE Proc. E (Comput. Digit. Tech.) 138 (3) (1991) 165–168.
[4] Z. Zhang, Q. Qi, N. Kumar, N. Chilamkurti, H.-Y. Jeong, A secure authentication scheme with anonymity for session initiation protocol using elliptic curve cryptography, Multimedia Tools Appl. 74 (10) (2015) 3477–3488.
[5] M. Aman, K.C. Chua, B. Sikdar, Mutual authentication in IoT systems using physical unclonable functions, IEEE Internet Things J. 4 (5) (2017) 1327–1340.
[6] Z. Ali, A. Ghani, I. Khan, S.A. Chaudhry, S.H. Islam, D. Giri, A robust authentication and access control protocol for securing wireless healthcare sensor networks, J. Inf. Secur. Appl. 52 (2020) 102502.
[7] A. Irshad, H. Naqvi, S. Ashraf Chaudhary, M. Usman, M. Shafiq, O. Mir, A. Kanwal, Cryptanalysis and improvement of a multi-server authenticated key agreement by Chen and Lee's scheme, Inf. Technol. Control 47 (3) (2018) 431–446.
[8] K. Mahmood, X. Li, S.A. Chaudhry, H. Naqvi, S. Kumari, A.K. Sangaiah, J.J. Rodrigues, Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure, Future Gener. Comput. Syst. 88 (2018) 491–500.
[9] A. Irshad, M. Usman, S.A. Chaudhry, H. Naqvi, M. Shafiq, A provably secure and efficient authenticated key agreement scheme for Energy Internet based Vehicle-to-Grid technology framework, IEEE Trans. Ind. Appl. (2020).
[10] S.A. Chaudhry, K. Yahya, F. Al-Turjman, M.-H. Yang, A secure and reliable device access control scheme for IoT based sensor cloud systems, IEEE Access 8 (2020) 139244–139254.
[11] M.L. Das, A. Saxena, V.P. Gulati, A dynamic ID-based remote user authentication scheme, IEEE Trans. Consum. Electron. 50 (2) (2004) 629–631.
[12] I.-E. Liao, C.-C. Lee, M.-S. Hwang, Security enhancement for a dynamic ID-based remote user authentication scheme, in: International Conference on Next Generation Web Services Practices, NWeSP 05, IEEE, 2005, pp. 4–pp.
[13] E.-J. Yoon, K.-Y. Yoo, Improving the dynamic ID-based remote mutual authentication scheme, in: OTM Confederated International Conferences" on the Move To Meaningful Internet Systems", Springer, 2006, pp. 499–507.
[14] Y.-y. Wang, J.-y. Liu, F.-x. Xiao, J. Dan, A more efficient and secure dynamic ID-based remote user authentication scheme, Comput. Commun. 32 (4) (2009) 583–585.
[15] F. Wen, X. Li, An improved dynamic ID-based remote user authentication with key agreement scheme, Comput. Electr. Eng. 38 (2) (2012) 381–387.
[16] Y.-F. Chang, W.-L. Tai, H.-C. Chang, Untraceable dynamic-identity-based remote user authentication scheme with verifiable password update, Int. J. Commun. Syst. 27 (11) (2014) 3430–3440.
[17] S. Kumari, M.K. Khan, X. Li, An improved remote user authentication scheme with key agreement, Comput. Electr. Eng. 40 (6) (2014) 1997–2012.
[18] S.A. Chaudhry, T. Shon, F. Al-Turjman, M.H. Alsharif, Correcting design flaws: An improved and cloud assisted key agreement scheme in cyber physical systems, Comput. Commun. 153 (2020) 527–537, http://dx.doi.org/10.1016/j.comcom.2020.02.025.
[19] S. Hussain, S.A. Chaudhry, Comments on 'biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment', IEEE Internet Things J. 6 (6) (2019) 10936–10940, http://dx.doi.org/10.1109/JIOT.2019.2934947.
[20] D. He, N. Kumar, J.-H. Lee, Privacy-preserving data aggregation scheme against internal attackers in smart grids, Wirel. Netw. 22 (2) (2016) 491–502.
[21] M.S. Farash, O. Nawaz, K. Mahmood, S.A. Chaudhry, M.K. Khan, A provably secure RFID authentication protocol based on elliptic curve for healthcare environments, J. Med. Syst. 40 (7) (2016) 165.
[22] M. Aman, B. Sikdar, ATT-auth: A hybrid protocol for industrial IoT attestation with authentication, IEEE Internet Things J. 5 (6) (2018) 5119–5131.
[23] S. Chaudhry, H. Alhakami, A. Baz, F. Al-Turjman, Securing demand response management: A certificate-based access control in smart grid edge computing infrastructure, IEEE Access 8 (2020) 101235–101243.
[24] F. Farivar, M.S. Haghighi, A. Jolfaei, M. Alazab, Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial IoT, IEEE Trans. Ind. Inf. 16 (4) (2020) 2716–2725.
[25] R. Vinayakumar, M. Alazab, S. Srinivasan, Q. Pham, S.K. Padannayil, K. Simran, A visualized botnet detection system based deep learning for the internet of things networks of smart cities, IEEE Trans. Ind. Appl. 56 (4) (2020) 4436–4456.
[26] M. Alazab, S. Huda, J. Abawajy, R. Islam, J. Yearwood, S. Venkatraman, R. Broadhurst, A hybrid wrapper-filter approach for malware detection, J. Netw. 9 (11) (2014) 2878–2891.
[27] C.-M. Chen, K.-H. Wang, K.-H. Yeh, B. Xiang, T.-Y. Wu, Attacks and solutions on a three-party password-based authenticated key exchange protocol for wireless communications, J. Ambient Intell. Hum. Comput. 10 (8) (2019) 3133–3142.

[28] S.D. Kaul, A.K. Awasthi, Security enhancement of an improved remote user authentication scheme with key agreement, Wirel. Pers. Commun. 89 (2) (2016) 621–637.
[29] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, M.T.M. Shalmani, On the power of power analysis in the real world: A complete break of the KeeLoq code hopping scheme, in: Annual International Cryptology Conference, Springer, 2008, pp. 203–220.
[30] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inform. Theory 29 (2) (1983) 198–208.
[31] X. Cao, S. Zhong, Breaking a remote user authentication scheme for multi-server architecture, IEEE Commun. Lett. 10 (8) (2006) 580–581.
[32] K. Mansoor, A. Ghani, S.A. Chaudhry, S. Shamshirband, S.A.K. Ghayyur, A. Mosavi, Securing IoT-based RFID systems: A robust authentication protocol using symmetric cryptography, Sensors 19 (21) (2019) 4752.
[33] C. Lin, D. He, N. Kumar, K.-K.R. Choo, A. Vinel, X. Huang, Security and privacy for the internet of drones: Challenges and solutions, IEEE Commun. Mag. 56 (1) (2018) 64–69.
[34] A. Ghani, K. Mansoor, S. Mehmood, S.A. Chaudhry, A.U. Rahman, M. Najmus Saqib, Security and key management in IoT-based wireless sensor networks: An authentication protocol using symmetric key, Int. J. Commun. Syst. 32 (16) (2019) e4139.
[35] Z. Ali, S.A. Chaudhry, M.S. Ramzan, F. Al-Turjman, Securing smart city surveillance: A lightweight authentication mechanism for unmanned vehicles, IEEE Access 8 (2020) 43711–43724.
[36] A hybrid deep learning image-based analysis for effective malware detection, J. Inf. Secur. Appl. 47 (2019) 377–389.
[37] C. Benzaid, K. Lounis, A. Al-Nemrat, N. Badache, M. Alazab, Fast authentication in wireless sensor networks, Future Gener. Comput. Syst. 55 (2016) 362–375.
[38] P. Kocher, J. Jaffe, B. Jun, P. Rohatgi, Introduction to differential power analysis, J. Cryptogr. Eng. 1 (1) (2011) 5–27.
[39] T.S. Messerges, E.A. Dabbish, R.H. Sloan, Examining smart-card security under the threat of power analysis attacks, IEEE Trans. Comput. 51 (5) (2002) 541–552.