Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **7**, Number **4**, November **2022**

An Efficient and Privacy Protection Authentication Protocol for Edge Computing

Yi Luo[1,2,*], Houpeng Hu[3], Bin Qian[1,2], Zhenghao Gao[3]
Yong Xiao[1,2], Jiaxiang Ou[3], Peilin He[3]

[1]Institute of Metrology Technology
Electric Power Research Institute, CSG, China

[2]Guangdong Provincial Key Laboratory of Intelligent Measurement and
Advanced Metering of Power Grid, China
luoyi_csg@outlook.com, qianbin@csg.cn, xiaoyong@csg.cn

[3]Guizhou Power Grid Co., LTD, China
huhp@gzsyy.csg.cn, zhgao_csg@163.com, oujx@gzsyy.csg.cn, 353868720@qq.com

*Corresponding author: Yi Luo

Received August 2, 2022, revised September 16, 2022, accepted October 23, 2022.

Abstract. *Due to resource constraints, cloud computing has high latency and network instability. Along with it, fog computing and edge computing to alleviate the problems faced by cloud computing. Fog computing and edge computing can process resources in a lower network structure, especially edge computing can be closer to the user layer. To ensure the security of network communication in the edge computing environment, we propose an efficient authentication protocol that protects users' privacy and can prove to be secure, which truly guarantees anonymity. The security analysis includes informal security analysis, Real-Or-Random (ROR) probability analysis and ProVerif verification tool. The performance analysis includes three aspects: security, computational cost and communication cost. By comparing with relevant protocols, it is proved that the proposed protocol not only has better security, but also ensures higher performance.*
Keywords: Edge computing, Fog computing, Anonymity, Security protocol

1. **Introduction.** With the rapid development of the Internet, various issues have been applied to the Internet of things [1, 2, 3, 4, 5] such as vehicle [6, 7, 8, 9], healthcare [10, 11, 12, 13, 14, 15, 16], and 5G [17, 18]. Due to resource constraints, cloud computing services [19, 20, 21, 22, 23, 24] are facing high latency and network instability. To improve the processing speed of user requests and the security of the network, the edge computing services [25, 26, 27, 28] place servers on the edge of the environment and store content as close as possible to the clients with requirements, so as to reduce the delay and improve the loading speed. In fact, the concepts of edge computing and fog computing [29, 30, 31] are very similar, and in some cases can even be interchangeable. But the main difference between them is the location of data processing. The data processing of fog computing is mainly carried out in the gateway or fog node. Edge computing services can place computing resources closer to devices or users for processing. In this context, edge computing technology has attracted more and more attention. Edge server can be applied in online banking, which ensures higher security and provides users with specific data through decentralized network model. In remote monitoring, the use of the edge server can ensure the centralization of requests and make them more effectively transmitted to

the source server. For autonomous vehicles, which require lower network latency during driving, the use of edge servers to transfer computing power to a closer edge can improve the safety of their work. Using edge servers to move the data center to the edge can offset the cost of cloud computing, ensure that the source server processes fewer requests, pays attention to specific requests, and responds to user requests faster. The edge server has various shapes and sizes, which makes it easier to install in the edge environment. Due to the advantages of edge server in the Internet of things, edge computing technology has become the research focus of scholars. The architecture of edge computing is shown in Figure 1.

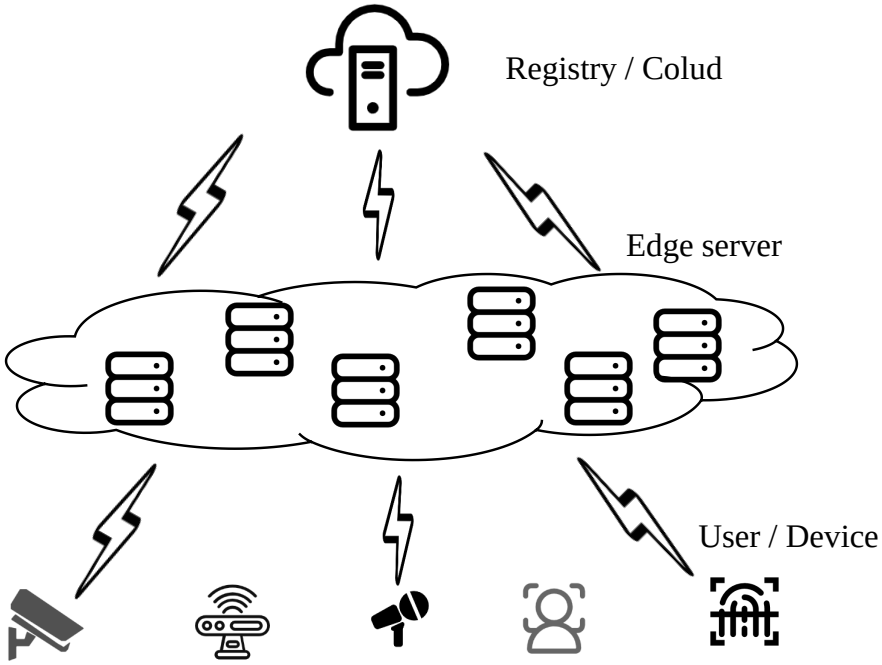

FIGURE 1. The architecture of edge computing

However, the distributed architecture of edge computing also increases the scope of attacks, making users or devices vulnerable to security attacks launched by malicious attackers. In addition, the highly dynamic environment at the edge of the network will also make the network more vulnerable and difficult to protect. For edge devices with limited resources, some current authentication and key agreement protocols are not fully applicable to edge computing architecture, so it is necessary to design secure and lightweight authentication and key agreement protocols for edge computing environment.

1.1. **Review of literature.** The development of fog computing and edge computing provides better services for the Internet of things. In order to ensure the quality of service of these two technologies, many authentication and key agreement protocols for fog servers or edge servers have been proposed in recent years.

In 2019, Waizd et al. [32] argued that fog computing inherits some security and privacy issues from cloud computing. Therefore, they designed a key management protocol suitable for the resource-constrained fog computing environment. In 2021, Ali et al. [33] pointed out that the protocol of Waizd et al. [32] was vulnerable to user identity tracking attacks and clogging attacks. Ali et al. improved the protocol of Waizd et al. [32], claiming to have corrected the security vulnerabilities in [32]. In 2021, Guo et al. [34]

proposed an authentication protocol that is convenient for mobile devices to switch communication in the fog computing environment. The authentication and key management phase of the protocol only involves users and fog nodes. Lee et al. [35] proposed a lightweight authentication protocol for inter-device communication based on fog computing. In the same year, Wu et al. [36] pointed out that some current schemes to apply fog nodes in social vehicle networking still have some challenges. Therefore, they proposed an authentication protocol for social networking of vehicles based on three-factor and fog nodes, and claimed that the protocol only uses lightweight computing and can provide better security. However, in 2022, Li et al. [37] found that Wu et al. 's protocol [36] was vulnerable to internal attacks and stolen smart card attacks, and could not guarantee perfect forward security. To make up for these security vulnerabilities, Li et al. proposed an authentication protocol for social networking of vehicles based on fog computing.

In 2018, Mahmood et al. [38] believed that the current relevant protocols could not guarantee the anonymity and reasonable security of smart meter facilities. They proposed an authentication protocol based on identity signature and bilinear pairings for smart grid edge computing facilities. In 2019, Wang et al. [39] pointed out that the authentication protocol in the smart grid environment does not support conditional anonymity and the key management is not flexible. Therefore, they proposed a blockchain-based anonymous and key management protocol for smart grid edge computing facilities to achieve secure communication between end users and edge servers. In the same year, Kaur et al. [40] proposed a mutual authentication protocol that can resist known attacks by using elliptic curve cryptography (ECC) and the Diffie-Hellman discrete logarithm problem. The protocol involves three communication entities: users, edge servers and trusted registries. In the authentication phase, users directly communicate with the server without the participation of a third party. Jia et al. [41] discussed the importance of achieving anonymity and non-traceability for users, and designed an anonymous authentication scheme for mobile edge computing. In 2020, Rostampour et al. [42] believed that it was still a great challenge to design a resource-saving and secure authentication protocol for IoT edge devices. They first analyzed the security of KS [43], CWS [44], KKD [45] and WCF [46], and found that these protocols were vulnerable to tracing attacks and man-in-the-middle attacks. To overcome these two security vulnerabilities, Rostampour et al. proposed an ECC-based authentication protocol for Internet edge devices, which only involves two entities, user and server. Deebak et al. [47] proposed a seamless and anonymous authentication protocol for mobile edge computing, which only involves mobile devices and cloud servers. In 2022, Zhang and Wei [48] proposed a lightweight and anonymous protocol for edge computing environment, but we found that their protocol could not provide anonymity. When the attacker obtains the information in the smart card, he can recover the r_i, and intercept the rid_i to calculate the identity. Further, off-line password guessing attacks, key disclosure attacks and perfect forward security can be launched.

1.2. **Our contribution.**

1. Reviewing and summarizing the authentication and key agreement protocols related to fog computing and edge computing in recent years, it is found that there are two major problems: high computing consumption and low security. We propose a provably secure and efficient authentication and key agreement protocol, which truly guarantees anonymity and resists other attacks.
2. Informal security analysis and formal security analysis are carried out for the proposed protocol. Informal security analysis includes anonymous, perfect forward security, impersonation attacks, internal attacks and other common attacks. Formal security analysis includes ROR probability analysis and ProVerif verification tool.

Table 1. Symbols and Descriptions

Symbol	Description
U_i	user
S_j	server
RA	registry
$\mathcal{A}$	attacker
x	the secret master key of the registry
T_i	timestamp
$E_k(\cdot)$	the symmetric encryption function
$D_k(\cdot)$	the symmetric decryption function
$H(\cdot)$	the hash function
$\|\|$	connect operation
$\oplus$	exclusive or operation

3. The security, computation cost and communication cost of the proposed protocol are analyzed. From these three aspects, the performance of the proposed protocol is compared with the related protocols, which proves that the proposed protocol has better performance.

2. **Proposed protocol.** Our efficient and anonymous protocol involves three entities: user U_i, edge server S_j and registration security center RA. The edge server only helps users and the security center transmit messages, and will not participate in user authentication. The protocol consists of three phases: user registration, server registration, authentication and key agreement. Table 1 shows the symbols used in the protocol.

2.1. **User registration phase.**

1. The user U_i selects his or her own identity ID_i and the corresponding password PW_i, chooses a random number n, and calculates $VPW_i = H(ID_i \oplus PW_i \oplus n)$, then sends $\{ID_i, VPW_i\}$ to RA via a secure channel.
2. After receiving the message, RA chooses a random number r_i and a pseudo identity VID_i, and calculates $A_i = H(ID_i \parallel VPW_i) \oplus H(VID_i \parallel ID_i \parallel x)$, $R_i = r_i \oplus H(VID_i \parallel VPW_i \parallel x)$. Then, RA stores $\{VID_i, ID_i, R_i\}$ in its own database and sends $\{A_i, R_i, VID_i\}$ to U_i.
3. After receiving the returned message, U_i calculates $n_1 = n \oplus H(ID_i \parallel PW_i \parallel VID_i)$, $V_i = H(ID_i \parallel VPW_i \parallel PW_i \parallel n)$, $R_u = R_i \oplus H(ID_i \parallel PW_i \parallel V_i)$. Then, U_i stores $\{R_u, VID_i, n_1, V_i, A_i\}$ in the smart card SC.

The user registration phase is shown in Figure 2.

2.2. **Server registration phase.**

1. The server S_j selects identity ID_j and a random number m, then sends $\{ID_j, m\}$ to RA via a secure channel.
2. After receiving the message, RA chooses a random number r_j and a pseudo identity VID_j, and calculates $A_j = H(ID_j \parallel m) \oplus H(VID_j \parallel ID_j \parallel x)$, $R_j = r_j \oplus H(VID_j \parallel m \parallel x)$. Then, RA stores $\{VID_j, ID_j, R_j\}$ in database and sends $\{A_j, R_j, VID_j\}$ to S_j.
3. After receiving the returned message, S_j calculates $R_s = R_j \oplus H(ID_j \parallel m)$. Then, S_j stores $\{VID_j, R_s, A_j\}$ in memory.

The server registration phase is shown in Figure 3.

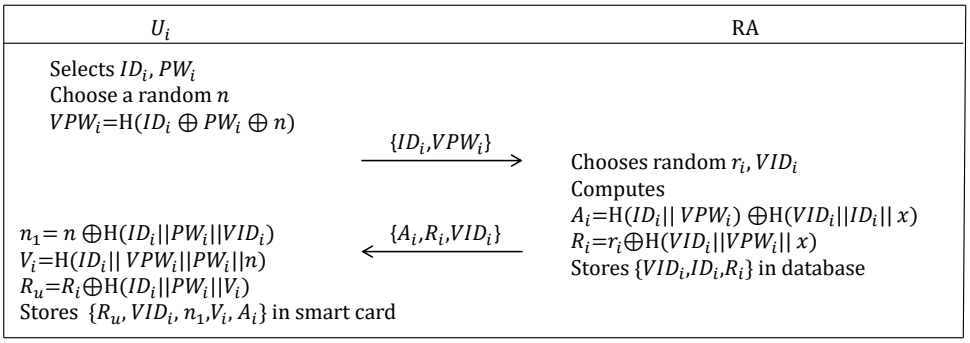

FIGURE 2. The user registration phase

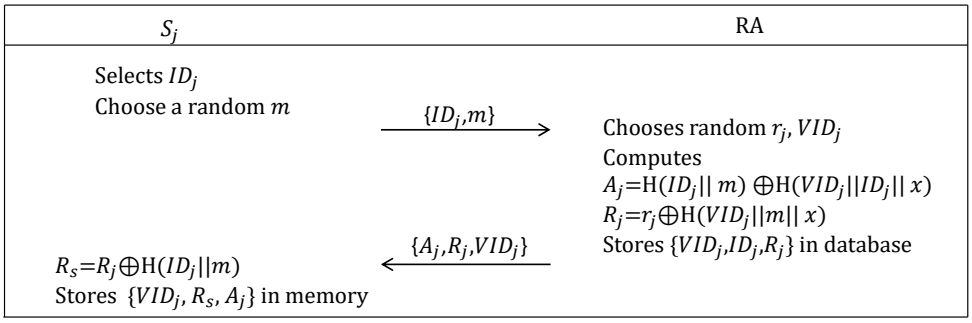

FIGURE 3. The server registration phase

2.3. **Authentication and key agreement phase.**

1. The user U_i inserts the smart card and inputs ID_i and PW_i. Then U_i computes $n = n_1 \oplus H(ID_i \parallel PW_i \parallel VID_i)$, $VPW_i = H(ID_i \oplus PW_i \oplus n)$, and checks $V_i \stackrel{?}{=} H(ID_i \parallel VPW_i \parallel PW_i \parallel n)$. If not, terminate the session. Otherwise, U_i selects a random number s_1 and a timestamp T_1, computes $V_1 = s_1 \oplus A_i \oplus H(ID_i \parallel VPW_i)$, $V_2 = H(s_1 \parallel ID_i \parallel VID_i \parallel T_1 \parallel V_1)$, and sends $M_1 = \{V_1, V_2, VID_i, T_1\}$ to S_j.
2. S_j checks whether T_1 is valid. If the times out, the communication is terminated. Otherwise, S_j chooses a random number s_2 and a timestamp T_2, computes $V_3 = s_2 \oplus A_j \oplus H(ID_j \parallel m)$, $V_4 = H(s_2 \parallel ID_j \parallel VID_j \parallel T_2 \parallel V_3)$, and sends $M_2 = \{V_1, V_2, V_3, V_4, VID_i, VID_j, T_1, T_2\}$ to RA.
3. RA checks whether T_2 is valid. If the times out, the communication is terminated. Otherwise, according to VID_i and VID_j, find $\{ID_i, R_i\}$ and $\{ID_j, R_j\}$ respectively. Then RA computes $r_i = R_i \oplus H(VID_i \parallel VPW_i \parallel x)$, $r_j = R_j \oplus H(VID_j \parallel m \parallel x)$, $s_1 = V_1 \oplus H(VID_i \parallel ID_i \parallel x)$, $s_2 = V_3 \oplus H(VID_j \parallel ID_j \parallel x)$, checks $V_2 \stackrel{?}{=} H(s_1 \parallel ID_i \parallel VID_i \parallel T_1 \parallel V_1)$ and $V_4 \stackrel{?}{=} H(s_2 \parallel ID_j \parallel VID_j \parallel T_2 \parallel V_3)$. If not, terminate the session. Otherwise, RA selects VID_i', VID_j', computes $A_i' = H(ID_i \parallel VPW_i) \oplus H(VID_i' \parallel ID_i \parallel x)$, $R_i' = r_i \oplus H(VID_i' \parallel VPW_i \parallel x)$, $A_j' = H(ID_j \parallel m) \oplus H(VID_j' \parallel ID_j \parallel x)$, $R_j' = r_j \oplus H(VID_j' \parallel m \parallel x)$, and updates $\{VID_i', ID_i, R_i'\}$ and $\{VID_j', ID_j, R_j'\}$ in the database respectively. Further, RA selects s_3, s_4, T_3, and computes $u = H(VID_i \parallel VPW_i \parallel x) \oplus s_3$, $V_5 = H(VID_i' \parallel ID_i \parallel s_1)$, $s = H(VID_j \parallel m \parallel x) \oplus s_4$, $V_6 = H(VID_j' \parallel ID_j \parallel s_2 \parallel T_3)$, $E_1 = E_{s_3 \oplus r_i \oplus H(VID_i \| VPW_i \| x)}(s_1, s_2, VID_i', VID_j', R_i', A_j', V_5)$, $E_2 =$

$E_{s_4 \oplus r_j \oplus H(VID_j \| m \| x)}(s_1, s_2, VID_i', VID_j', R_j', A_j', V_6)$, $SK_r = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$. Finally, RA sends $M_3 = \{E_1, E_2, u, s, T_3\}$ to S_j.

4. S_j checks whether T_3 is valid. If the times out, the communication is terminated. Otherwise, S_j computes $(s_1, s_2, VID_i', VID_j', R_j', A_j', V_6) = D_{s \oplus R_s \oplus H(ID_j \| m)}(E_2)$, $V_6' = H(VID_j' \parallel ID_j \parallel s_2 \parallel T_3)$, and checks $V_6' \stackrel{?}{=} V_6$. If not, terminate the session. Otherwise, S_j selects T_4, computes $R_s' = R_j' \oplus H(ID_j \parallel m)$, $SK_s = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$, and updates $\{VID_j', R_s', A_j'\}$ in memory. Finally, S_j sends $M_4 = \{E_1, u, T_4\}$ to U_i.
5. U_i checks whether T_4 is valid. If the times out, the communication is terminated. Otherwise, U_i computes $(s_1, s_2, VID_i', VID_j', R_i', A_i', V_5) = D_{u \oplus R_u \oplus H(ID_i \| PW_i \| V_i)}(E_1)$, $V_5' = H(VID_i' \parallel ID_i \parallel s_1)$, and checks $V_5' \stackrel{?}{=} V_5$. Then U_i computes $R_u' = R_i' \oplus H(ID_i \parallel PW_i \parallel V_i)$, $SK_u = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$, and updates $\{R_u', VID_i', A_i'\}$ in the smart card.

The authentication and key agreement phase is shown in Figure 4.

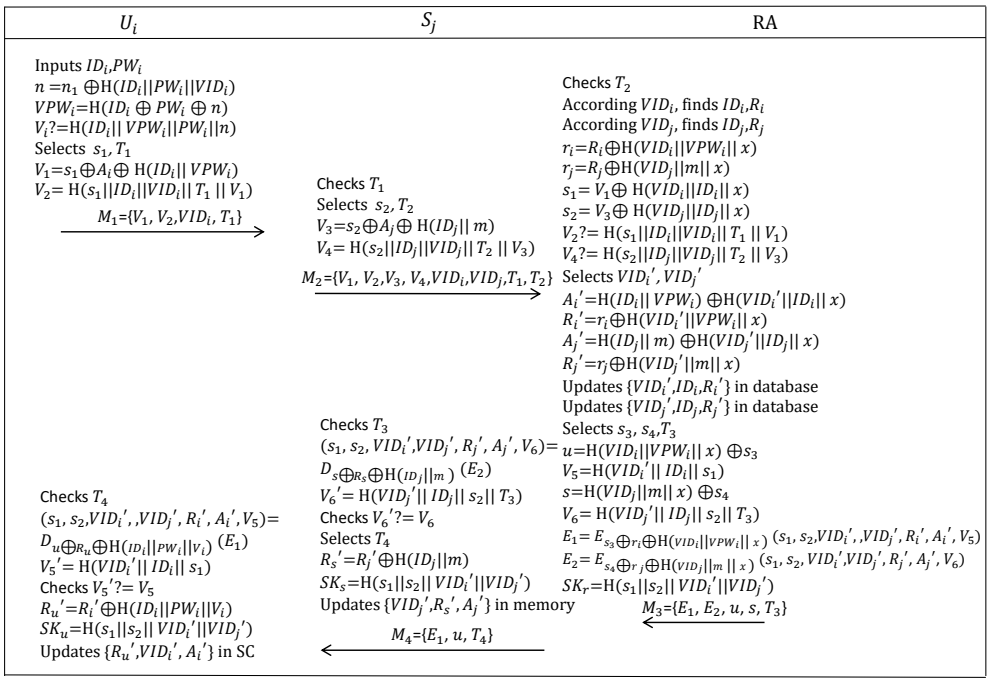

FIGURE 4. The authentication and key agreement phase

3. Security analysis.

3.1. Formal security analysis.

3.1.1. *ROR probability analysis.* In the Real-Or-Random (ROR) model [49, 50, 46, 51], $\mathcal{A}$ uses different queries such as $Execute$, $Send$, $Hash$, $Corrupt$, and $Test$ to obtain the probability of success. $Execute$, $Send$, $Hash$, $Corrupt$, and $Test$ query respectively represent the attacker passively captures the messages transmitted by the public channel, actively intercepts the messages transmitted by the public channel, outputs the corresponding hash value, captures the secret value, and flips a coin to judge the random result. In the proposed protocol, the corresponding communication instances of U_i, S_j and RA in ROR model can be defined as Π_U^i, Π_S^j, Π_{RA}^i represents the i-th instance of U_i, the j-th instance of S_j, and the k-th instance of RA, respectively.

Definition 3.1. *Symmetric encryption and decryption algorithm (Ω). k_1, k_2, $\cdots$, k_n represent the key used for encryption or decryption, and E_{k_1}, E_{k_2}, $\cdots$, E_{k_n} are the encrypted values of n keys respectively. In polynomial time ξ, the probability of $\mathcal{A}$ breaking the secret parameter k is $Adv_{\mathcal{A}}^{\Omega,k}(\xi) = |2Pr[\mathcal{A} \leftarrow E_{k_1}; (b_0, b_1) \leftarrow \mathcal{A}; \alpha \leftarrow 0, 1; \beta \leftarrow E_{k_1}(b_\alpha) : \mathcal{A}(\beta) = \alpha] - 1|$.*

Theorem 3.1. *Assuming that $\mathcal{A}$ can break the proposed protocol $\mathcal{P}$ by executing queries in polynomial time ξ, the probability is $Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) \leq q_{send}/2^{l-1} + q_{hash}^2/2^{l-1} + 2Adv_{\mathcal{A}}^{\Omega,k}(\xi)$, where q_{hash} is the number of $Hash$ queries, q_{send} is the number of $Send$ queries, and l is the length of the transmitted text.*

Proof: The game sequences GM_0-GM_5 are used for probability analysis and proof. $Succ_{\mathcal{A}}^{GM_n}(\xi)$ is the probability that $\mathcal{A}$ will succeed in the game GM_n. The proof is as follows.

GM_0: Toss a coin to start the game, and $\mathcal{A}$ does not execute the query. The probability of $\mathcal{A}$ breaking $\mathcal{P}$ is

$$Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) = |2Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)] - 1|. \tag{1}$$

GM_1: Run $Execute$ query. $\mathcal{A}$ passively gets the messages M_1-M_4, and there is no other operation. At this point, the probability of GM_1 is equal to GM_0, that is

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

GM_2: Execute $Send$ query. $\mathcal{A}$ intercepts and forges the messages M_1-M_4 and attempts to get a response. According to Zipf's law [52], the probability of $\mathcal{A}$ breaking $\mathcal{P}$ is

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

GM_3: Execute $Hash$ query. $\mathcal{A}$ enters a string, and attempts to output the correct hash value. According to the birthday paradox, the probability of $\mathcal{A}$ breaking $\mathcal{P}$ is

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

GM_4: Execute $Corrupt$ query. $\mathcal{A}$ obtains a secret value in the communication instance, such as x, s_1, s_2, etc., and attempts to launch known session-specific temporary information attacks or verify perfect forward secrecy. However, in both cases, E_1 or E_2 needs to be decrypted to calculate the session key $SK_u = SK_s = SK_r = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$. In other words, the probability of GM_4 is equal to that of the symmetric encryption and decryption algorithm, that is,

$$|Pr[Succ_{\mathcal{A}}^{GM_4}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_3}(\xi)]| \leq Adv_{\mathcal{A}}^{\Omega,k}(\xi). \tag{5}$$

GM_5: Execute $H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$ query to attempt to launch the session key disclosure attacks. The probability of $\mathcal{A}$ breaking $\mathcal{P}$ is

$$|Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_4}(\xi)]| \leq q_{hash}^2/2^{l+1}. \tag{6}$$

The probability of correctly guessing the session key is equal to that of not, so,

$$Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)] = 1/2. \tag{7}$$

According to formula (1)-(7), we can get

$$\begin{aligned} 1/2Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) &= |Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)] - 1/2| \\ &= |Pr[Succ_{\mathcal{A}}^{GM_0}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)]| \\ &= |Pr[Succ_{\mathcal{A}}^{GM_1}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)]| \\ &\leq \sum_{i=0}^{4} |Pr[Succ_{\mathcal{A}}^{GM_{i+1}}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_i}(\xi)]| \\ &= q_{send}/2^l + q_{hash}^2/2^l + Adv_{\mathcal{A}}^{\Omega,k}(\xi). \end{aligned} \tag{8}$$

Further, we can get $Adv_{\mathcal{A}}^{\mathcal{P}}(\xi) \leq q_{send}/2^{l-1} + q_{hash}^2/2^{l-1} + 2Adv_{\mathcal{A}}^{\Omega,k}(\xi)$.

3.1.2. *ProVerif.* ProVerif automatic verification tool [53, 54, 55, 56] can describe relevant primitives in cryptography and judge whether the set event occurs by executing code. Simulate all processes of the proposed protocol, start the reasoning algorithm, and verify the security of the whole protocol. The whole simulation process is divided into declaration part, event part, query part, process part and main function part. The contents of each part are described in detail below in combination with the proposed protocol.

In the declaration part, the channels, variables and functions in the protocol are defined, as shown in Figure 5. (a). In the event and query parts, the protocol is formalized. $\mathcal{A}$ queries the session key and the logical sequence of each event, and finally outputs whether the query can be successfully executed. As shown in Figure 5. (b).

```
(* * * * * * * * * * * * * * * * * * channe* * * * * * * * * * * * * * * * *)
free ch :channel. (* public channel *)
free sch: channel [private]. (* * *secure channel, used for registering * * *)
(* * * * * * * * * * * * * * * * * * *shared keys* * * * * * * * * * * * * * * *)
free SKu:bitstring [private].
free SKs:bitstring [private].
free SKr:bitstring [private].
(* * * * * * * * * * * * * * * * * * *constants * * * * * * * * * * * * * * * *)
free x:bitstring [private]. (* the RA's secret key *)
(* * * * * * * * * * * *functions & reductions & equations* * * * * * * * *)
fun H(bitstring) :bitstring. (*  hash function *)
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 senc(bitstring,bitstring):bitstring. (* symmetric encryption *)
reduc forall m:bitstring, key:bitstring; sdec(senc(m,key),key)=m.
```

(a)Definitions

```
(* * * * * * * * * * * * * * * * * * *queries* * * * * * * * * * * * * * * * * )
query attacker(SKu).
query attacker(SKs).
query attacker(SKr).
query inj-event(UserAuthed()) ==> inj-event(UserStarted()).
query inj-event(RAAcServer()) ==> inj-event(RAAcUser()).
query inj-event(ServerAcRA()) ==> inj-event(RAAcServer()).
query inj-event(UserAcRA()) ==> inj-event(ServerAcRA()).

(* * * * * * * * * * * * * * * * * * * event* * * * * * * * * * * * * * * * *)

event UserStarted().
event UserAuthed().
event RAAcUser().
event RAAcServer().
event ServerAcRA().
event UserAcRA().
```

(b)Events and queries

FIGURE 5. Definitions and queries

The process part describes the detailed steps of each entity in the protocol, including the definition of new parameters, sending and receiving messages, etc. The main function part calls all the process parts to end the whole program. As shown in the Figure 6. (a) and Figure 6. (b).

Finally, when executing the program, the system calls the main function and finally outputs the result, as shown in the Figure 6. (c).

3.2. **Informal security analysis.**

3.2.1. *Anonymity.* When users U_i and servers S_j register with the registry RA, RA will correspondingly generate virtual identity VID_i, VID_j for mutual authentication later. In the public channel of authentication and key agreement phase, only VID_i and VID_j are transmitted between U_i and S_j. The real identity ID_i and ID_j are confidential, and the attacker $\mathcal{A}$ cannot recover the real identity by other means. Therefore, our proposed protocol guarantees anonymity.

```
(* * * * * * * * * * * * * User's process * * * * * * * * * * * *)
let ProcessUi =
new IDi:bitstring;
new PWi:bitstring;  new n:bitstring;
let VPWi=H(xor(xor(IDi,PWi),n)) in
out(sch,(IDi,VPWi));
in(sch,(xAi:bitstring,xRi:bitstring,xVIDi:bitstring));
let n1=xor(n,H(con(con(IDi,PWi),xVIDi))) in
let Vi=H(con(con(con(IDi,VPWi),PWi),n)) in
let Ru=xor(xRi,H(con(con(IDi,PWi),Vi))) in
!(event UserStarted();
let n=xor(n1,H(con(con(IDi,PWi),xVIDi))) in
let VPWi=H(xor(xor(IDi,PWi),n)) in
let Vi'=H(con(con(con(IDi,VPWi),PWi),n)) in
if Vi'=Vi then
new s1:bitstring;    new T1:bitstring;
let V1=xor(xor(s1,xAi),H(con(IDi,VPWi))) in
let V2=H(con(con(con(con(s1,IDi),xVIDi),T1),V1))
inout(ch,(V1,V2,xVIDi,T1)); (* ----- authentication ----- *)
event UserAuthed();
in(ch,(xE1:bitstring,xu:bitstring,xT4:bitstring));
let ku=xor(xor(xu,Ru),H(con(con(IDi,PWi),Vi'))) in
let (s1:bitstring,s2:bitstring,VIDi':bitstring,VIDj':bitstring,
Ri':bitstring,Ai':bitstring,V5:bitstring)=sdec(xE1,ku) in
let V5'=H(con(con(VIDi',IDi),s1)) in
if V5=V5'  then event UserAcRA();
let Ru'=xor(Ri',H(con(con(IDi,PWi),Vi'))) in
let SKu=H(con(con(con(s1,s2),VIDi'),VIDj')) in
0 (* ----- authentication ----- *)).
(* * * * * * * * * * * * * Server's process * * * * * * * * * * * *)
let ProcessSj=
new IDj:bitstring;    new m:bitstring;
out sch,(IDj,m));in(sch,(yAj:bitstring,yRj:bitstring,yVIDj:bitstring));
let Rs=xor(yRj,H(con(IDj,m))) in
!(in(ch,(yV1:bitstring,yV2:bitstring,yVIDi:bitstring,yT1:bitstring));n
ew s2:bitstring;       new T2:bitstring;
let V3=xor(xor(s2,yAj),H(con(IDj,m))) in
let V4=H(con(con(con(con(s2,IDj),yVIDj),T2),V3)) in
out(ch,(yV1,yV2,yVIDi,yT1,V3,V4,yVIDj,T2));
in(ch,(yE1:bitstring,yE2:bitstring,yu:bitstring,ys:bitstring,yT3:bitstr
ing));
let ks=xor(xor(ys,Rs),H(con(IDj,m))) in
let (s1:bitstring,s2:bitstring,VIDi':bitstring,VIDj':bitstring,
Rj':bitstring,Aj':bitstring,V6:bitstring)=sdec(yE2,ks) in
let V6'=H(con(con(con(VIDj',IDj),s2),yT3)) in
if V6'=V6   then event ServerAcRA();
new T4:bitstring;
let Rs'=xor(Rj',H(con(IDj,m))) in
let SKs=H(con(con(con(s1,s2),VIDi'),VIDj')) in
out(ch,(yE1,yu,T4));
0).
```

(a) Process

```
(* * * * * * * * * * * * * *RA's process* * * * * * * * * * * *)
let UiReg =
in(sch,(zIDi:bitstring,zVPWi:bitstring));
new ri:bitstring;       new VIDi:bitstring;
let Ai=xor(H(con(zIDi,zVPWi)),H(con(con(VIDi,zIDi),x))) in
let Ri=xor(ri,H(con(con(VIDi,zVPWi),x))) in
out(sch,(Ai,Ri,VIDi));
0.(* -----Ui registration ----- *)

let SjReg =
in (sch,(zIDj:bitstring,zm:bitstring));
new rj:bitstring;       new VIDj:bitstring;
let Aj=xor(H(con(zIDj,zm)),H(con(con(VIDj,zIDj),x))) in
let Rj=xor(rj,H(con(con(VIDj,zm),x))) in
out(sch,(Aj,Rj,VIDj));
0.(* -----Sj registration ----- *)

let RAAuth =
in(ch,(zV1:bitstring,zV2:bitstring,zVIDi:bitstring,zT1:bitstring,zV3
:bitstring,zV4:bitstring,zVIDj:bitstring,zT2:bitstring));
new zVPWi:bitstring; new Ri:bitstring; new zm:bitstring;
new Rj:bitstring; new zIDi:bitstring; new  zIDj:bitstring;
let ri=xor(Ri,H(con(con(zVIDi,zVPWi),x))) in
let rj=xor(Rj,H(con(con(zVIDj,zm),x))) in
let s1=xor(zV1,H(con(con(zVIDi,zIDi),x))) in
let s2=xor(zV3,H(con(con(zVIDj,zIDj),x))) in
let V2'=H(con(con(con(con(s1,zIDi),zVIDi),zT1),zV1)) in
if V2'=zV2  then event RAAcUser(); (*-------RA verifies Ui-------*)
let V4'=H(con(con(con(con(s2,zIDj),zVIDj),zT2),zV3)) in
if V4'=zV4  then  event RAAcServer(); (*-----RA verifies Sj-----*)
new VIDi':bitstring;  new VIDj':bitstring;
let Ai'=xor(H(con(zIDi,zVPWi)),H(con(con(zVIDi,zIDi),x))) in
let Ri'=xor(ri,H(con(con(zVIDi,zVPWi),x))) in
let Aj'=xor(H(con(zIDj,zm)),H(con(con(zVIDj,zIDj),x))) in
let Rj'=xor(rj,H(con(con(zVIDj,zm),x))) in
new s3:bitstring;     new s4:bitstring;     new T3:bitstring;
let u=xor(H(con(con(zVIDi,zVPWi),x)),s3) in
let V5=H(con(con(VIDi',zIDi),s1)) in
let s=xor(H(con(con(zVIDj,zm),x)),s4) in
let V6=H(con(con(con(VIDj',zIDj),s2),T3)) in
let E1=senc((s1,s2,VIDi',VIDj',Ri',Ai',V5),xor(xor(s3,ri),
H(con(con(zVIDi,zVPWi),x)))) in
let E2=senc((s1,s2,VIDi',VIDj',Rj',Aj',V6),
xor(xor(s4,rj),H(con(con(zVIDj,zm),x)))) in
let SKr=H(con(con(con(s1,s2),VIDi'),VIDj')) in
out(ch,(E1,E2,u,s,T3));
0.

let ProcessRA  = UiReg | SjReg | RAAuth.
(* ----- Main ----- *)
process    (!ProcessUi | ! ProcessSj| !ProcessRA)
```

(b) Process

```
(* * * * * * * * * * * * * * * * * * results* * * * * * * * * * * * * * )
1-- RESULT not attacker(SKu[]) is true.
2-- RESULT not attacker(SKs[]) is true.
3-- RESULT not attacker(SKr[]) is true.
4-- RESULT inj-event(UserAuthed) ==> inj-event(UserStarted) is true.
5-- RESULT inj-event(RAAcServer) ==> inj-event(RAAcUser) is true.
6-- RESULT inj-event(ServerAcRA) ==> inj-event(RAAcServer) is true.
7-- RESULT inj-event(UserAcRA) ==> inj-event(ServerAcRA) is true.
```

(c) Results

FIGURE 6. Processes and result

3.2.2. *Perfect forward secrecy.* In this protocol, the session key of the U_i, S_j and RA is $SK_u = SK_s = SK_r = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$. If $\mathcal{A}$ obtains the master key x of RA and attempts to get $\{s_1, s_2, VID_i', VID_j'\}$ by decrypting E_1 or E_2. Then, $\mathcal{A}$ also needs to obtain $\{s_3, R_i, VPW_i\}$ or $\{s_4, R_j, m\}$, however, these parameters are confidential and cannot be obtained by $\mathcal{A}$. Therefore, $\mathcal{A}$ cannot recover the session key, and the proposed protocol provides perfect forward secrecy.

3.2.3. *Stolen smart card attacks.* If $\mathcal{A}$ steals the smart card SC, he can get the information $\{R_u, VID_i, n_1,$
$V_i, A_i\}$. $\mathcal{A}$ attempts to recover the secret values R_i and n, but also needs to get the identity ID_i and password PW_i, VPW_i, and these parameters are confidential. Therefore, even if $\mathcal{A}$ steals the data in the smart card, it will not pose a threat to the whole protocol. The proposed protocol is resistant to stolen smart card attacks.

3.2.4. *Off-line password guessing attacks.* If $\mathcal{A}$ gets the information $\{VID_i, n_1, V_i\}$ in the smart card, verify $V_i \stackrel{?}{=} H(ID_i \parallel VPW_i \parallel PW_i \parallel n)$ to guess the password PW_i, where $VPW_i = H(ID_i \oplus PW_i \oplus n)$, and $\{ID_i, PW_i, n\}$ are unknown. It is obviously impossible to guess the three parameters ID_i, PW_i and n at the same time. Therefore, the proposed protocol resists off-line password guessing attacks.

3.2.5. *User impersonation attacks.* Suppose $\mathcal{A}$ attempts to forge $M_1 = \{V_1, V_2, VID_i, T_1\}$ and launch a user impersonation attack disguised as the legitimate user. Then $\mathcal{A}$ selects s_1', T_1', and calculates the values $V_1' = s_1' \oplus A_i \oplus H(ID_i \parallel VPW_i)$ and $V_2' = H(s_1' \parallel ID_i \parallel VID_i \parallel T_1' \parallel V_1')$. However, $\{A_i, ID_i, PW_i, VPW_i\}$ are confidential, and $\mathcal{A}$ cannot pass the subsequent verification. Therefore, the protocol provided resists user impersonation attacks.

3.2.6. *Server impersonation attacks.* Suppose $\mathcal{A}$ attempts to forge $M_2 = \{V_1, V_2, V_3, V_4, VID_i, VID_j, T_1, T_2\}$ and launch a server impersonation attack disguised as the legitimate server. Then $\mathcal{A}$ selects s_2', T_2', and calculates the values $V_3' = s_2' \oplus A_j \oplus H(ID_j \parallel m)$ and $V_4' = H(s_2' \parallel ID_j \parallel VID_j \parallel T_2' \parallel V_3')$. However, $\{A_j, ID_j, m\}$ are confidential, and $\mathcal{A}$ cannot pass the subsequent verification. Therefore, the protocol provided resists server impersonation attacks.

3.2.7. *Insider attacks.* Suppose $\mathcal{A}$ obtains user information $\{VID_i, ID_i, R_i\}$ or server information $\{VID_j$,
$ID_j, R_j\}$ stored in the database of RA and attempts to calculate the session key $SK_u = SK_s = SK_r = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$, where $\{s_1, s_2, VID_i', VID_j'\}$ are encrypted transmission. To calculate the decryption key, $\mathcal{A}$ need to obtain s_3 or s_4, however, s_3 and s_4 are confidential. Therefore, the proposed protocol resists internal attacks.

3.2.8. *Known session-specific temporary information attacks.* In this protocol, the session key of the U_i, S_j and RA is $SK_u = SK_s = SK_r = H(s_1 \parallel s_2 \parallel VID_i' \parallel VID_j')$. Suppose $\mathcal{A}$ gets the random number s_1 generated by U_i, the other three parameters $\{s_2, VID_i', VID_j'\}$ are encrypted transmission. To calculate the decryption key, $\mathcal{A}$ need to obtain s_3 or s_4, however, s_3 and s_4 are confidential. Suppose $\mathcal{A}$ obtains the random number s_2 generated by S_j, and the result is the same as above. Therefore, the proposed protocol resists known session-specific temporary information attacks.

4. **Performance analysis.** In this section, the proposed protocol is compared with Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48] respectively in terms of security, computational cost and communication cost.

4.1. **Security comparison.** Table 2 shows the security comparison between the proposed protocol and Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48], where A1 means anonymity or untraceability, A2 means perfect forward secrecy, A3 indicates stolen smart card attacks, A4 indicates off-line password guessing attacks, A5 indicates user impersonation attacks, A6 indicates server impersonation attacks, A7 indicates internal attacks, A8 indicates known session-specific temporary information attacks, A9 indicates clogging attacks. The "$\surd$" indicates that the security feature can be realized, and the "$\times$" indicates that the security feature cannot be realized. According to table 2, we can find that Waizd et al. 's protocol [32] cannot provide untraceability and cannot resist clogging attacks. Jia et al. 's protocol [41] cannot provide untraceability, and is vulnerable to stloen smart card attacks, user impersonation attacks and internal attacks. Zhang et al. 's protocol [48] cannot guarantee anonymity and perfect forward

Table 2. Security Comparison

	Waizd et al. [32]	Jia et al. [41]	Zhang et al. [48]	Our protocol
A1	× [33]	× [48]	×	√
A2	√	√	×	√
A3	√	× [48]	√	√
A4	√	√	×	√
A5	√	× [48]	√	√
A6	√	√	√	√
A7	√	× [48]	√	√
A8	√	√	√	√
A9	× [33]	-	-	-

Table 3. Computational Cost Comparison

	User	Edge server /Fog node	Registry	Device	Total
Waizd et al. [32]	$21T_h + T_a + 2T_m + T_f$	$16T_h + T_a + 3T_m$	-	$17T_h$	$51.916\,ms$
Jia et al. [41]	$5T_h + 3T_a + 6T_m + T_{ex}$	$5T_h + 3T_a + 5T_m + T_{map}$	-	-	$116.34\,ms$
Zhang et al. [48]	$8T_h + 3T_s$	$6T_h + 2T_s$	$15T_h + 2T_s$	-	$82.716\,ms$
Our protocol	$8T_h + T_s$	$5T_h + T_s$	$17T_h + 2T_s$	-	$47.32\,ms$

security, and cannot resist off-line password guessing attacks. Our proposed protocol has better security.

4.2. **Computational cost comparison.** Table 3 shows the computational cost comparison between the proposed protocol and Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48]. Here, the Windows 10 operating system with Inter(R) Core(TM) I5-8500 CPU @ 3.00Hz and 8G memory is used. The development software is InterliiJ idea 2019.3, call the Java pairing library, signature library and symmetric encryption and decryption function. After ten experiments, calculate the average values $T_h = 0.004ms$, $T_m = 8.6ms$, $T_f = T_m$, $T_a = 0.05ms$, $T_s = 11.8ms$, $T_{map} = 10.6ms$, $T_{ex} = 10.8ms$, where T_h, T_m, T_f, T_a, T_s, T_{map}, T_{ex} respectively represent the operation time of hash operation, elliptic curve scalar point multiplication, fuzzer, point addition operation, symmetric encryption and decryption, bilinear pair and exponential operation. Here, only the computational costs of authentication and key agreement phases are compared. According to table 3, we get the results that our proposed protocol has the lowest computational cost. Figure 7 shows the computational cost comparison between our protocol and Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48].

4.3. **Communication cost comparison.** Table 4 shows the communication cost comparison between the proposed protocol and Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48]. Assume that the points of the elliptic curve occupy 512 bits, hash and symmetric encryption and decryption occupy 256 bits respectively, and timestamp and random number occupy 64 bits respectively. Only the communication costs of authentication and key agreement phases are compared here.

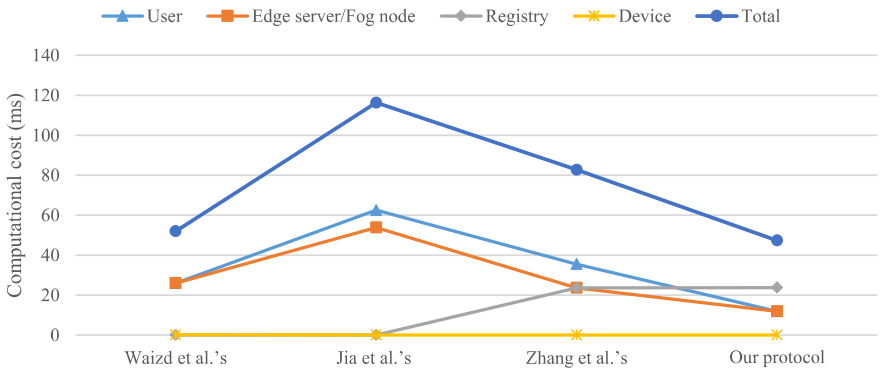

FIGURE 7. Computational cost comparison

TABLE 4. Communication Cost Comparison

	Rounds	Communication cost
Waizd et al. [32]	3	4800 bits
Jia et al. [41]	2	1984 bits
Zhang et al. [48]	5	3904 bits
Our protocol	4	3584 bits

According to each specific protocol, we can get that the message transmitted in Waizd et al.'s protocol [32] contains 4 points, 3 timestamps, 10 hash values, and the communication cost is 4800 bits. The message transmitted in Jia et al.'s protocol [41] contains 3 points, 1 random number, 2 timestamps, 1 hash value, and the communication cost is 1984 bits. The message transmitted in Zhang et al.'s protocol [48] contains 6 random numbers, 3 timestamps, 8 hash values and 5 encrypted values, and the communication cost is 3904 bits. The message transmitted in our protocol contains 3 random numbers, 5 timestamps, 9 hash values and 3 encrypted values, and the communication cost is 3584 bits. According to table 4, we can find that our scheme is only higher than Jia et al.'s protocol [41], but the calculation cost of [41] is twice that of our protocol, and it has security vulnerabilities. Therefore, the overall performance of our proposed protocol is good. Figure 8 shows the communication cost comparison between our protocol and Waizd et al.'s protocol [32], Jia et al.'s protocol [41] and Zhang et al.'s protocol [48].

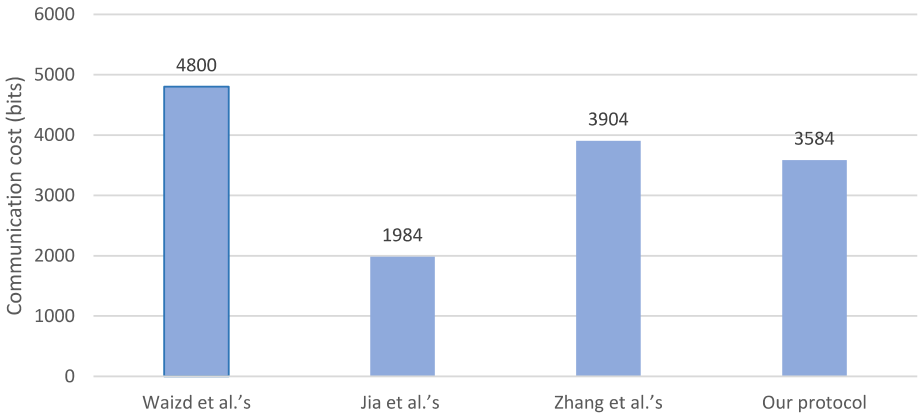

FIGURE 8. Communication cost comparison

5. **Conclusions.** According to the characteristics of edge computing and fog computing, this paper proposes an authentication and key agreement protocol that can protect user privacy and prove security. Then the security is proved by using informal security analysis, ROR model and ProVerif verification tool. It shows that the scheme has correct logicality, stable security, and integrity of authentication process. Finally, the security, computational cost and communication cost of the proposed protocol and related protocols are evaluated through performance analysis. The results show that the proposed protocol has the best overall performance and is very suitable for edge computing and fog computing environments.

Acknowledgment. This research was partially supported by National Key Research and Development Program of China (Grant Nos. 2019YFE0118700), Science and Technology Project of China Southern Power Grid Corporation (Grant No. 066600KK52200016).

REFERENCES

[1] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "On the security of a new ultra-lightweight authentication protocol in iot environment for rfid tags," *The Journal of Supercomputing*, vol. 74, no. 1, pp. 65–70, 2018.

[2] T.-Y. Wu, C.-M. Chen, K.-H. Wang, and J. M.-T. Wu, "Security analysis and enhancement of a certificateless searchable public key encryption scheme for iiot environments," *IEEE Access*, vol. 7, pp. 49232–49239, 2019.

[3] L. Kang, R.-S. Chen, N. Xiong, Y.-C. Chen, Y.-X. Hu, and C.-M. Chen, "Selecting hyper-parameters of gaussian process regression based on non-inertial particle swarm optimization in internet of things," *IEEE Access*, vol. 7, pp. 59504–59513, 2019.

[4] S. Shen, Y. Yang, and X. Liu, "Toward data privacy preservation with ciphertext update and key rotation for iot," *Concurrency and Computation: Practice and Experience*, p. e6729, 2021.

[5] G. Liu, Y. Zhu, S. Xu, Y.-C. Chen, and H. Tang, "Pso-based power-driven x-routing algorithm in semiconductor design for predictive intelligence of iot applications," *Applied Soft Computing*, vol. 114, p. 108114, 2022.

[6] T.-Y. Wu, Z. Lee, L. Yang, and C.-M. Chen, "A provably secure authentication and key exchange protocol in vehicular ad hoc networks," *Security and Communication Networks*, vol. 2021, p. 9944460, 2021.

[7] T. Wu, X. Guo, Y. Chen, S. Kumari, and C. Chen, "Amassing the security: An enhanced authentication protocol for drone communications over 5g networks," *Drones*, vol. 6, no. 1, p. 10, 2021.

[8] H. Xiong, J. Chen, Q. Mei, and Y. Zhao, "Conditional privacy-preserving authentication protocol with dynamic membership updating for vanets," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 2089–2104, 2022.

[9] T.-Y. Wu, X. Guo, Y.-C. Chen, S. Kumari, and C.-M. Chen, "Sgxap: Sgx-based authentication protocol in iov-enabled fog computing," *Symmetry*, vol. 14, no. 7, p. 1393, 2022.

[10] Y. Yang, X. Zheng, W. Guo, X. Liu, and V. Chang, "Privacy-preserving smart iot-based healthcare big data storage and self-adaptive access control system," *Information Sciences*, vol. 479, pp. 567–592, 2019.

[11] Y. Yang, X. Zheng, W. Guo, X. Liu, and V. Chang, "Privacy-preserving fusion of iot and big data for e-health," *Future Generation Computer Systems*, vol. 86, pp. 1437–1455, 2018.

[12] A. Kumari, V. Kumar, M. Y. Abbasi, S. Kumari, P. Chaudhary, and C.-M. Chen, "Csef: cloud-based secure and efficient framework for smart medical system using ecc," *IEEE Access*, vol. 8, pp. 107838–107852, 2020.

[13] T.-Y. Wu, T. Wang, Y.-Q. Lee, W. Zheng, S. Kumari, and S. Kumar, "Improved authenticated key agreement scheme for fog-driven iot healthcare system," *Security and Communication Networks*, vol. 2021, p. 6658041, 2021.

[14] T.-Y. Wu, L. Yang, J.-N. Luo, and J. Ming-Tai Wu, "A provably secure authentication and key agreement protocol in cloud-based smart healthcare environments," *Security and Communication Networks*, vol. 2021, p. 2299632, 2021.

[15] T.-Y. Wu, L. Yang, Q. Meng, X. Guo, and C.-M. Chen, "Fog-driven secure authentication and key exchange scheme for wearable health monitoring system," *Security and Communication Networks*, vol. 2021, p. 8368646, 2021.

[16] T.-Y. Wu, Q. Meng, L. Yang, S. Kumari, and M. P. Nia, "Amassing the security: An enhanced authentication and key agreement protocol for remote surgery in healthcare environment," *Computer Modeling in Engineering and Sciences*, vol. 134, no. 1, pp. 317–341, 2023.

[17] T.-Y. Wu, Z. Lee, M. S. Obaidat, S. Kumari, S. Kumar, and C.-M. Chen, "An authenticated key exchange protocol for multi-server architecture in 5g networks," *IEEE Access*, vol. 8, pp. 28096–28108, 2020.

[18] L. Yang, Y.-C. Chen, and T.-Y. Wu, "Provably secure client-server key management scheme in 5g networks," *Wireless Communications and Mobile Computing*, vol. 2021, p. 4083199, 2021.

[19] T.-Y. Wu, Y.-M. Tseng, S.-S. Huang, and Y.-C. Lai, "Non-repudiable provable data possession scheme with designated verifier in cloud storage systems," *IEEE Access*, vol. 5, pp. 19333–19341, 2017.

[20] L. Kang, R.-S. Chen, Y.-C. Chen, C.-C. Wang, X. Li, and T.-Y. Wu, "Using cache optimization method to reduce network traffic in communication systems based on cloud computing," *IEEE Access*, vol. 7, pp. 124397–124409, 2019.

[21] T.-Y. Wu, X. Fan, K.-H. Wang, C.-F. Lai, N. Xiong, and J. M.-T. Wu, "A dna computation-based image encryption scheme for cloud cctv systems," *IEEE Access*, vol. 7, pp. 181434–181443, 2019.

[22] R. Tso, K. Huang, Y.-C. Chen, S. M. M. Rahman, and T.-Y. Wu, "Generic construction of dual-server public key encryption with keyword search on cloud computing," *IEEE Access*, vol. 8, pp. 152551–152564, 2020.

[23] T.-Y. Wu, Q. Meng, S. Kumari, and P. Zhang, "Rotating behind security: A lightweight authentication protocol based on iot-enabled cloud computing environments," *Sensors*, vol. 22, no. 10, p. 3858, 2022.

[24] T.-Y. Wu, L. Wang, X. Guo, Y.-C. Chen, and S.-C. Chu, "Sakap: Sgx-based authentication key agreement protocol in iot-enabled cloud computing," *Sustainability*, vol. 14, no. 17, p. 11054, 2022.

[25] X. Chen, J. Zhang, B. Lin, Z. Chen, K. Wolter, and G. Min, "Energy-efficient offloading for dnn-based smart iot systems in cloud-edge environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 3, pp. 683–697, 2021.

[26] T.-Y. Wu, Q. Meng, L. Yang, X. Guo, and S. Kumari, "A provably secure lightweight authentication protocol in mobile edge computing environments," *The Journal of Supercomputing*, vol. 78, pp. 13893–13914, 2022.

[27] J. Zhang, M. Li, Z. Chen, and B. Lin, "Computation offloading for object-oriented applications in a uav-based edge-cloud environment," *The Journal of Supercomputing*, vol. 78, no. 8, pp. 10829–10853, 2022.

[28] Q. Mei, H. Xiong, Y.-C. Chen, and C.-M. Chen, "Blockchain-enabled privacy-preserving authentication mechanism for transportation cps with cloud-edge computing," *IEEE Transactions on Engineering Management*, 2022. 10.1109/TEM.2022.3159311.

[29] Y. Yang, X. Liu, W. Guo, X. Zheng, C. Dong, and Z. Liu, "Multimedia access control with secure provenance in fog-cloud computing networks," *Multimedia Tools and Applications*, vol. 79, no. 15, pp. 10701–10716, 2020.

[30] T.-Y. Wu, Z. Lee, L. Yang, J.-N. Luo, and R. Tso, "Provably secure authentication key exchange scheme using fog nodes in vehicular ad hoc networks," *The Journal of Supercomputing*, vol. 77, no. 7, pp. 6992–7020, 2021.

[31] U. Chatterjee, S. Ray, M. K. Khan, M. Dasgupta, and C.-M. Chen, "An ecc-based lightweight remote user authentication and key management scheme for iot communication in context of fog computing," *Computing*, vol. 104, pp. 1359–1395, 2022.

[32] M. Wazid, A. K. Das, N. Kumar, and A. V. Vasilakos, "Design of secure key management and user authentication scheme for fog computing services," *Future Generation Computer Systems*, vol. 91, pp. 475–492, 2019.

[33] Z. Ali, S. A. Chaudhry, K. Mahmood, S. Garg, Z. Lv, and Y. B. Zikria, "A clogging resistant secure authentication scheme for fog computing services," *Computer Networks*, vol. 185, p. 107731, 2021.

[34] Y. Guo and Y. Guo, "Fogha: An efficient handover authentication for mobile devices in fog computing," *Computers & Security*, vol. 108, p. 102358, 2021.

[35] T.-F. Lee and W.-Y. Chen, "Lightweight fog computing-based authentication protocols using physically unclonable functions for internet of medical things," *Journal of Information Security and Applications*, vol. 59, p. 102817, 2021.

[36] T.-Y. Wu, X. Guo, L. Yang, Q. Meng, and C.-M. Chen, "A lightweight authenticated key agreement protocol using fog nodes in social internet of vehicles," *Mobile Information Systems*, vol. 2021, p. 3277113, 2021.

[37] Z. Li, Q. Miao, S. A. Chaudhry, and C.-M. Chen, "A provably secure and lightweight mutual authentication protocol in fog-enabled social internet of vehicles," *International Journal of Distributed Sensor Networks*, vol. 18, no. 6, p. 15501329221104332, 2022.

[38] K. Mahmood, X. Li, S. A. Chaudhry, H. Naqvi, S. Kumari, A. K. Sangaiah, and J. J. Rodrigues, "Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure," *Future Generation Computer Systems*, vol. 88, pp. 491–500, 2018.

[39] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain-based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1984–1992, 2019.

[40] K. Kaur, S. Garg, G. Kaddoum, M. Guizani, and D. N. K. Jayakody, "A lightweight and privacy-preserving authentication protocol for mobile edge computing," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2019.

[41] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing," *IEEE Systems Journal*, vol. 14, no. 1, pp. 560–571, 2019.

[42] S. Rostampour, M. Safkhani, Y. Bendavid, and N. Bagheri, "Eccbap: A secure ecc-based authentication protocol for iot edge devices," *Pervasive and Mobile Computing*, vol. 67, p. 101194, 2020.

[43] S. Kalra and S. K. Sood, "Secure authentication scheme for iot and cloud servers," *Pervasive and Mobile Computing*, vol. 24, pp. 210–223, 2015.

[44] C.-C. Chang, H.-L. Wu, and C.-Y. Sun, "Notes on "secure authentication scheme for iot and cloud servers"," *Pervasive and Mobile Computing*, vol. 38, pp. 275–278, 2017.

[45] S. Kumari, M. Karuppiah, A. K. Das, X. Li, F. Wu, and N. Kumar, "A secure authentication scheme based on elliptic curve cryptography for iot and cloud servers," *The Journal of Supercomputing*, vol. 74, no. 12, pp. 6428–6453, 2018.

[46] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "A secure authentication scheme for internet of things," *Pervasive and Mobile Computing*, vol. 42, pp. 15–26, 2017.

[47] B. D. Deebak, F. Al-Turjman, and L. Mostarda, "Seamless secure anonymous authentication for cloud-based mobile edge computing," *Computers & Electrical Engineering*, vol. 87, p. 106782, 2020.

[48] Z. Yukun and W. Wenxue, "Lightweight anonymous authentication and key agreement protocols for mobile edge computing," *Peer-to-Peer Networking and Applications*, vol. 15, pp. 1994–2006, 2022.

[49] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *Journal of the ACM (JACM)*, vol. 51, no. 4, pp. 557–594, 2004.

[50] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *International workshop on public key cryptography*, pp. 65–84, Springer, 2005.

[51] T.-Y. Wu, L. Yang, Z. Lee, S.-C. Chu, S. Kumari, and S. Kumar, "A provably secure three-factor authentication protocol for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 2021, p. 5537018, 2021.

[52] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017.

[53] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," *ACM Sigplan Notices*, vol. 36, no. 3, pp. 104–115, 2001.

[54] B. Blanchet, "A computationally sound mechanized prover for security protocols," *IEEE Transactions on Dependable and Secure Computing*, vol. 5, no. 4, pp. 193–207, 2008.

[55] T.-Y. Wu, L. Yang, Z. Lee, C.-M. Chen, J.-S. Pan, and S. Islam, "Improved ecc-based three-factor multiserver authentication scheme," *Security and Communication Networks*, vol. 2021, p. 6627956, 2021.

[56] T.-Y. Wu, Y.-Q. Lee, C.-M. Chen, Y. Tian, and N. A. Al-Nabhan, "An enhanced pairing-based authentication scheme for smart grid communications," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2021. https://doi.org/10.1007/s12652-020-02740-2.