SCIENCE CHINA
Information Sciences

March 2020, Vol. 63 130103:1–130103:15
https://doi.org/10.1007/s11432-019-9922-x

• RESEARCH PAPER •
Special Focus on Security and Privacy in Blockchain-based Applications

# A privacy preserving two-factor authentication protocol for the Bitcoin SPV nodes

Lu ZHOU[1], Chunpeng GE[2*] & Chunhua SU[1]

[1]*Division of Computer Science, University of Aizu, Aizuwakamatsu* 9658580*, Japan;*
[2]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing* 210000*, China*

Received 24 April 2019/Accepted 17 June 2019/Published online 10 February 2020

**Abstract** In the Bitcoin network, the simplified payment verification protocol (SPV) enables a lightweight device such as a mobile phone to participate in the bitcoin network without needed to download and store the whole Bitcoin blocks. A Bitcoin SPV node initiates and verifies transactions of the Bitcoin network through the Bitcoin wallet software which is deployed on a resource constrained device such as a mobile phone. Thus, the security of the wallet is critical for the SPV nodes as it may affect the security of user's cryptocurrencies. However, there are some concerns about the security flaws within the SPV nodes which could lead to significant economic losses. Most of these vulnerabilities can be resolved by employing a secure user authentication protocol. Over the years, researchers have engaged in designing a secure authentication protocol. However, most proposals have security flaws or performance issues. Recently, Park et al. proposed a two-party authenticated key exchange protocol for the mobile environment. They claimed that their protocol is not only secure against various attacks but also can be deployed efficiently. However, after a thorough security analysis, we find that the Park et al.'s protocol is vulnerable to user forgery attack, smart card stolen attack and unable to provide user anonymity. To enhance security, we proposed an efficient and secure user authentication protocol for the SPV nodes in the mobile environment which can fulfill all the security requirements and has provable security. Additionally, we provide performance analysis which shows our proposed protocol is efficient for the SPV nodes in the Bitcoin network.

**Keywords** SPV nodes, secure authentication, two-factor, Bitcoin

**Citation** Zhou L, Ge C P, Su C H. A privacy preserving two-factor authentication protocol for the Bitcoin SPV nodes. Sci China Inf Sci, 2020, 63(3): 130103, https://doi.org/10.1007/s11432-019-9922-x

## 1 Introduction

The Bitcoin is the most successful cryptocurrency and its capitalization has achieved about $100 billion [1]. In the Bitcoin network, there are two types of nodes, the full nodes, and SPV nodes. A full node stores all the blocks that contain block headers and block bodies. While an SPV node, which is always a lightweight device (e.g., a mobile phone) only stores the block headers. The SPV nodes can issue and verify transactions by relying on the SPV protocol. When an SPV node issues a transaction in the Bitcoin network, it should sign the transaction with his private key. However, the private is a 32 bytes lengthen meaningless string that is always hard to remember [2]. In order to manage the private keys, an SPV node always uses a Bitcoin wallet to store his private keys and issues transactions with a mobile phone.

However, the mobile devices are more vulnerable due to the constrained resources and lacking protection mechanism. There have been an array of threats that take advantage of numerous vulnerabilities commonly found in such devices. Those vulnerabilities can be a result of inadequate technical controls or poor security practices of consumers. There are some vulnerabilities that worth mention.

* Corresponding author (email: gecp@nuaa.edu.cn)

 

• **Cryptographic algorithms vulnerabilities:** The authentication protocols such as factor-based authentication [3–10] are easy to disrupt.

• **Software vulnerabilities:** Due to the software vulnerabilities of the mobile software such as the vulnerable library and operation system, the password may be leaked.

• **Hardware flaws:** The hardware authentication manner maybe simulated by some external equipments such as the "Dolphin Sound Attack" [11].

Consider the special environment of mobile devices, the factor-based protocols seem to be the most promising one to deploy and the two-factor authentication protocol is the most promising representative. The two-factor protocols can give considerably high security while not require large computational overhead. Thus, many two-factor protocols designed for mobile devices have been proposed. Recently, Park et al. [12] proposed a two-factor user authentication protocol. They claimed that their protocol is provably secure and can resist various security attacks. They also said the protocol can provide user anonymity. However, after a throughout security analysis, we found that the protocol of Part et al. is vulnerable to user forgery attack, smart card stolen attack and cannot provide user anonymity. To enhance security, we propose an efficient and secure two-factor user authentication protocol which is provably secure and is resistant to various known security attacks.

## 2 Review of literature

In the last decade, people tend to use mobile devices to access the Internet. While bringing a lot of conveniences, there are many obstacles that hinder its deployment. Thus, scholars proposed various authentication protocols which are capable of resisting security attacks, achieving mutual authentication and key agreement.

Among these proposals, there are some protocols worth mention. In 2012, He et al. [13] proposed an authentication protocol based on ECC which can provide known session key security, the perfect forward secrecy, the no key-compromise impersonation, the no unknown key-share and the no key control. The authors also claimed that their protocol can perform efficiently in mobile devices.

At the same time, Wu et al. [14] proposed a protocol offering medical information service based on mobile devices. The proposed protocol uses a pre-computing technique and thus avoid time-consuming computations during communication. However, later He et al. [15] pointed out that Wu et al.'s protocol is vulnerable to impersonation attack and insider attack. Thus, they come up with a newly improved protocol which can eliminate the weakness they found in the previous protocol. Later in that year, Wei et al. [16] found that both He et al. and Wu et al.'s protocols are failed to achieve two-factor authentication. To enhance security, they proposed their design which satisfies the security requirements of two-factor authentication.

In 2015, Wang et al. [17] designed a practical anonymous two-factor authentication scheme. They have revisited two foremost proposals, i.e., Tsai et al. [18]'s scheme and Li et al. [19]'s scheme, and made the first step towards understanding the underlying evaluation metric for anonymous two-factor authentication. Round the same time, Memon et al. [20] proposed a protocol for location-based services using asymmetric key cryptography. However, later in 2016, Reddy et al. [21] spotted that their protocol has various limitations such as vulnerable to key compromised impersonation attack, insecure password changing phase, imperfect mutual authentication, and vulnerable to insider attack. In 2017, Chaudhry et al. [22] proposed a protocol using ECC which is privacy preserving and efficient. Sequentially, Feng et al. [23] proposed an ideal lattice-based anonymous authentication protocol for mobile devices. The authors claimed that their protocol will be secure in the post-quantum era.

In 2017, Qi and Chen [24] proposed a two party authentication key exchange protocols with the purpose of achieving secure communication in client-server architectures in the mobile environment. However, unlike what the authors claimed, Part et al. [12] found the Qi and Chen's protocol is vulnerable to impersonation, off-line password guessing, password change, privileged insider attacks and cannot provide user anonymity as well. Thus, Part et al. proposed a new authentication protocol in order to cover all

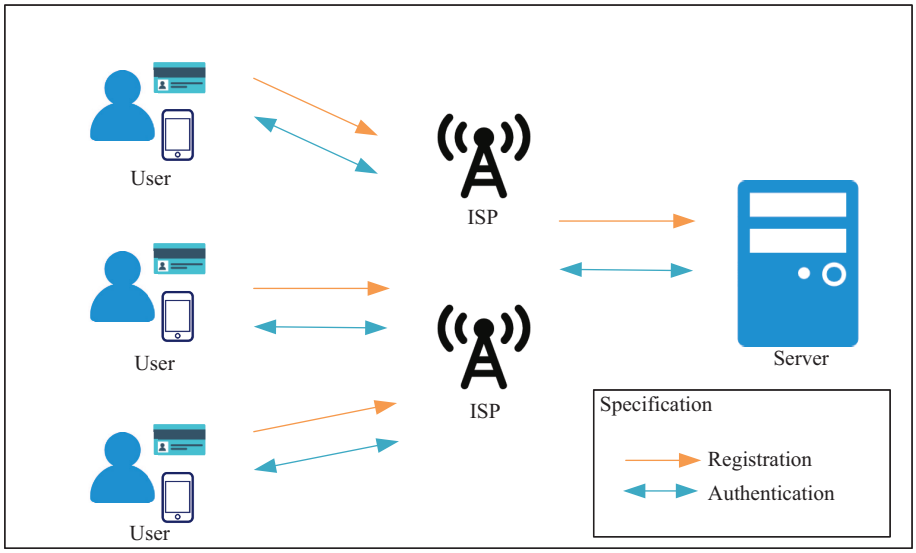


**Figure 1** (Color online) Network model.

vulnerabilities. Unfortunately, after a thorough security analysis, we found that Park et al.'s protocol is insecure against user forgery attack, smart card stolen attack and cannot provide user anonymity. To overcome these security weaknesses, we propose an improved two-factor user authentication protocol which can not only fulfill all the security requirements, but also efficient for real-world deployment.

## 3 Mathematical preliminaries

In this section, we list some mathematical problems which are known to be hard to solve.

• **Elliptic curve computational Diffie Hellman (EC-CDH) problem:** Given a elliptic curve $E$, a generator $P$, $x, y \in Z_p$, it is easy to calculate $xyP$. However, given $xP$, $yP$, it is computational hard to calculate $xyP$.

• **Elliptic curve discrete logarithm (ECDLP) problem:** Given a elliptic curve $E$. It is easy to calculate $Q = kP$ with $k \in Z_p$ and $P \in E$. However, it is computational hard to calculate corresponding $k$ such that $Q = kP$.

## 4 System framework and security model

### 4.1 Network framework

In this subsection, we will present the network model for our protocol. As shown in Figure 1, there are two kinds of participants in our protocol: (1) user ($U_i$) and (2) server ($S$). In order to get service from the server, the user first needs to register him/her self in the server. Later, when the user requires service, he/she has to pass the authenticate phase.

### 4.2 Security requirements

In this subsection, we will present some security requirements which an authentication protocol in the mobile environment should have.

**(1) Mutual authentication:** It is a basic security requirement which requires participants in the protocol can authenticate each other.

**(2) Forward secrecy:** This requirement that means even one or more participants have lost their long-term private keys, does not enable $\mathcal{A}$ to obtain the current session key.

**(3) User anonymity:** This requirement is aimed at protecting users. $\mathcal{A}$ cannot link two transcripts from different sessions to the same user.

**(4) Key agreement:** It is a basic requirement for user authentication protocol which requires two parties can generate the same session key.

**(5) Resistant to various security attacks:** It is a security requirement for this protocol which means the protocol should have the ability to resistant various security attacks, such as user forgery attack, server forgery attack, off-line password-guessing attack, and reply attack.

### 4.3 Security model

In this subsection, we will specify the threat model which is considered in most authentication protocols. The ability of adversary $\mathcal{A}$ can be described as follows:

- $\mathcal{A}$ has complete control over the public network, which indicates that $\mathcal{A}$ can intersect, modify, reply, delete any message in the authentication phase. Besides, $\mathcal{A}$ also has the ability to rearrange the order of messages.
- $\mathcal{A}$ can steal or copy the user's smart card and extract the security parameters stored in it.
- $\mathcal{A}$ may possess old long-term or short term private key, old session keys as well.
- $\mathcal{A}$ can crack the user's ID and password by the off-line password-guessing attack or other security attacks [25].
- $\mathcal{A}$ may be a vicious insider or a privileged user.

In our authentication protocol, we consider two parties: (1) server $S$ and (2) user $U_i$ in $U$. $\Pi_S^t$ and $\Pi_{U_i}^r$ denotes instances $t$ and $r$ of $U_i$ and $S$. The adversary $\mathcal{A}$ can interact with both of them within a polynomial time which allows us to perform real-time attacks on our protocol. According to many other published studies [26–28], we define some oracles which may be useful.

- Execute($\Pi^t, \Pi^r$): This query simulates the passive attacks which enable $\mathcal{A}$ to obtain any message from two honest instances $\Pi^t$ and $\Pi^r$ translated via public network.
- Reveal($\Pi^t$): This query reveals the current session key SK generated by $\Pi^t$ to $\mathcal{A}$.
- Send($\Pi^t, m$): This query simulates active attacks which enable $\mathcal{A}$ to send message $m$ to instance $\Pi^t$. In reply, $\mathcal{A}$ will get a response message from $\Pi^t$.
- CorruptSmartCard($\Pi_{U_i}^t$): This query simulates that $\mathcal{A}$ gets a smart card and extracts secret parameters from it.
- Test($\Pi^t$): This query determines whether the proposed protocol reaches semantic secure, which follows the indistinguishability in RO (random oracle) model. When this query is executed, an unbiased coin $c$ is tossed. If $c = 1$, $\Pi^t$ outputs session key SK, otherwise, it outputs random key drawn from the session-key domain.

### 4.4 Contribution

The contribution of this paper can be described as follows:

- We provide a security analysis for Park et al.'s protocol. We have shown that their protocol is insecure against user forgery attack, smart card stolen attack and cannot provide user anonymity.
- We propose an efficient and secure two-factor user authentication protocol which is provably secure and can resist various attacks.
- We provide a performance analysis which proves that our protocol is efficient and can be deployed in a real industrial environment.

## 5 Organization

This paper is organized as follows: In Section 2, we present some important studies related to our research. In Section 3, we provide some preliminary involved in our protocol. In Section 4, we provide system framework and security model. Then, we present Park et al.'s protocol in Section 6 and a security analysis for their protocol in Section 7. In Section 8, we provide our protocol and give a security analysis

**Table 1** Symbols were used in the protocol of Park et al. and our protocol

| Symbols | Description |
|---|---|
| $\mathcal{A}$ | An attacker |
| $U_i$ | $i$-th of user |
| $\mathrm{ID}_i$ | Identity of $U_i$ |
| $\mathrm{PW}_i$ | Password of $U_i$ |
| $S$ | The server |
| $d_S$ | Private key of the server |
| $Q_S$ | Public key of the server |
| $P$ | Elliptic curve point |
| $h(\cdot), H(\cdot)$ | Secure Hash functions |
| $\Pi$ | Our protocol |
| kdf | Secure one-way key derivation function |
| $\oplus$ | Exclusive-OR operation |
| $\|\|$ | Concatenation operation |

in Section 9 which proves our protocol is provably secure and can withstand various attacks. Later, in Section 10, we present a performance analysis which shows our protocol has low computational and communication costs and is suitable for the real environment. In Section 11, we conclude our paper.

## 6 Review of Park et al.'s protocol

The protocol of Park et al. has 3 parses: (1) user registration parse, (2) authentication and key agreement parse, and (3) password change phase.

There are two participants in this protocol, i.e., (1) the user $U_i$ and (2) the server $S$. Besides, we will present some symbols we used frequently in Table 1.

### 6.1 User registration parse

This parse is taken place between $U_i$ and server $S$ and can be described as follows:

(1) $U_i$ chooses his/her $\mathrm{ID}_i$ and password $\mathrm{PW}_i$, generates two random numbers $a_i$ and $b_i$, computes $\mathrm{RPW} = h(\mathrm{ID}_i||\mathrm{PW}_i)$, $v = \mathrm{RPW} \oplus a_i$, $c = h(\mathrm{ID}_i||\mathrm{PW}_i||a_i)$.

(2) $U_i$ sends $\{\mathrm{ID}_i, \mathrm{RPW} \oplus b_i\}$ to server $S$ via a secret channel.

(3) $S$ checks if $\mathrm{ID}_i$ and $h(\mathrm{ID}_i)$ exist, if that is true, $S$ asks $U_i$ to choose another identity. $S$ computes $l = H(d_S) \oplus (\mathrm{RPW} \oplus b_i) \oplus h(d_S \oplus \mathrm{ID}_i)$, stores $\mathrm{ID}_i, h(d_S||\mathrm{ID}_i)$ in the database.

(4) $S$ sends $l$ to $U_i$ through a secret channel.

(5) $U_i$ computes $l' = l \oplus b_i = H(d_S) \oplus \mathrm{RPW} \oplus h(d_S \oplus \mathrm{ID}_i)$ and stores $l', v, c$ in smart card or mobile device.

### 6.2 Authentication and key agreement parse

Before user $U_i$ can gain service from server $S$, they need to authenticate each other and generate the session key. As presented in Figure 2, the details of this parse can be described as follows:

(1) $U_i$ inputs his/her $\mathrm{ID}_i^*$ and $\mathrm{PW}_i^*$, computes $\mathrm{RPW}^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*)$, $a_i^* = v \oplus \mathrm{RPW}^*$, $c_i^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*||a_i^*)$. Then smart card checks if $c_i^* = c_i$. If the equation holds, smart card proceeds the process, otherwise, it rejects the authentication request.

(2) $U_i$ generates random number $r_u$ from $Z_N^*$ and current timestamp $T_1$, computes $R_u = r_u P$, $R = r_u Q_S$, $\mathrm{CID}_i = l' \oplus \mathrm{RPW} = h(d_S) \oplus h(d_S||\mathrm{ID}_i)$, $\mathrm{Auth}_u = h(\mathrm{ID}_i||R||\mathrm{CID}_i||T_1)$. $U_i$ sends $M_1 = \{\mathrm{Auth}_u, \mathrm{CID}_i, R_u, T_1\}$ to server $S$ via public channel.

(3) After receiving $M_1$, $S$ first check the timestamp $T_1$. If the timestamp is valid, $S$ proceeds the process, otherwise, it terminates the authentication request. Then $S$ computes $h(d_S||\mathrm{ID}_i) = \mathrm{CID}_i \oplus H(d_S)$, retrieves $\mathrm{ID}_i$ from database based on $h(d_S||\mathrm{ID}_i)$, computes $R^* = d_S R_u$, $\mathrm{Auth}_u^* = h(\mathrm{ID}_i||R^*||\mathrm{CID}_i||T_1)$.

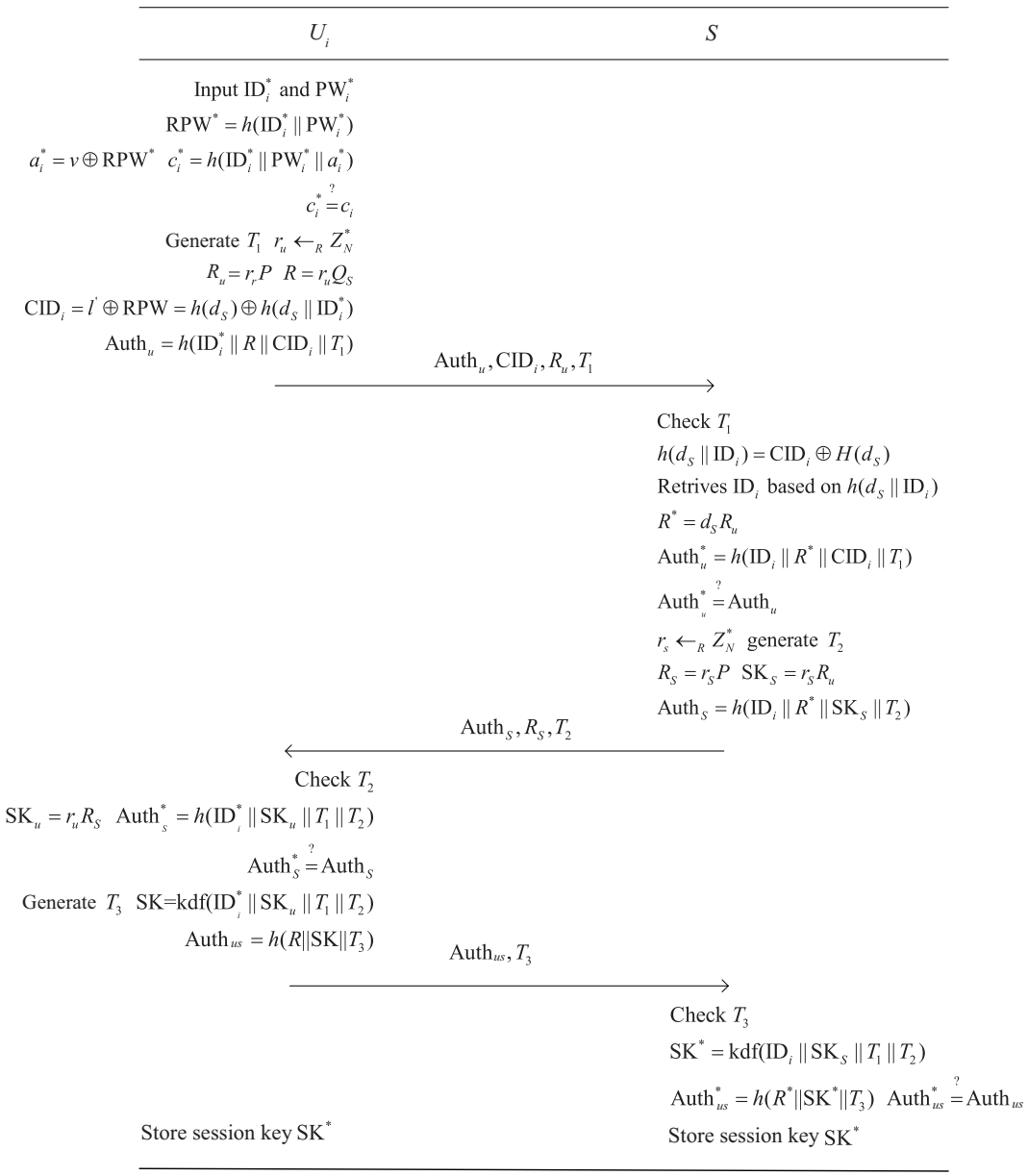


**Figure 2** Authentication and key agreement parse for Park et al.'s protocol

$S$ validates the equation $\mathrm{Auth}_u^* = \mathrm{Auth}_u$, if it holds, $S$ proceeds the authentication procedure, otherwise, $S$ terminates the process.

(4) $S$ generates random number $r_S$ from $Z_N^*$ and current timestamp $T_2$, computes $R_S = r_S P$, $\mathrm{SK}_S = r_S R_u$, $\mathrm{Auth}_S = h(\mathrm{ID}_i || R^* || \mathrm{SK}_S || T_2)$, $M_2 = \{\mathrm{Auth}_S, R_S, T_2\}$. Then, $S$ sends $M_2$ to $U_i$ via public channel.

(5) After receiving $M_2$, $U_i$ checks if the timestamp $T_2$ is valid, and computes $\mathrm{SK}_u = r_u R_S$, $\mathrm{Auth}_S^* = h(\mathrm{ID}_i || R || \mathrm{SK}_u || T_2)$. $U_i$ checks if the equation $\mathrm{Auth}_S^* = \mathrm{Auth}_S$ and if it is true, $U_i$ authenticates server $S$, otherwise, it terminates the authentication.

(6) $U_i$ generates current timestamp $T_3$, computes the session key $\mathrm{SK} = \mathrm{kdf}(\mathrm{ID}_i || \mathrm{SK}_u || T_1 || T_2)$, $\mathrm{Auth}_{us} = h(R || \mathrm{SK} || T_3)$, $M_3 = \{\mathrm{Auth}_{us}, T_3\}$. Then, $U_i$ sends $M_3$ to $S$ and stores the session key SK.

(7) After receiving $M_3$, $S$ checks the timestamp $T_3$. If the timestamp is valid, $S$ computes session key $\mathrm{SK}^* = \mathrm{kdf}(\mathrm{ID}_i || \mathrm{SK}_S || T_1 || T_2)$, $\mathrm{Auth}_{us}^* = h(R^* || \mathrm{SK}^* || T_3)$. Then $S$ checks the equation $\mathrm{Auth}_{us}^* = \mathrm{Auth}_{us}$. If the equation holds, $S$ authenticates $U_i$ as a legitimate user and stores the session key $\mathrm{SK}^*$.

### 6.3 Password change parse

This parse takes care of the situation when user wants to change his/her password and the details of this parse can be described as follows.

(1) $U_i$ inputs his/her identity $\mathrm{ID}_i^*$ and password $\mathrm{PW}_i^*$.

(2) Smart card or mobile device computes $\mathrm{RPW}^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*)$, $a_i^* = v \oplus \mathrm{RPW}^*$, $c^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*||a_i^*)$. Then smart card or mobile device checks if $c^* = c$ and if it is true, smart card authenticates $U_i$ and asks $U_i$ to enter new password $\mathrm{PW}^{\mathrm{new}}$.

(3) Smart card computes $\mathrm{RPW}^{\mathrm{new}} = h(\mathrm{ID}_i^*||\mathrm{PW}^{\mathrm{new}})$, $v^{\mathrm{new}} = \mathrm{RPW}^{\mathrm{new}} \oplus a_i^*$, $c^{\mathrm{new}} = h(\mathrm{ID}_i^*||\mathrm{PW}^{\mathrm{new}}||a_i^*)$, $l'^{\mathrm{new}} = l' \oplus \mathrm{RPW} \oplus \mathrm{RPW}^{\mathrm{new}} = H(d_S) \oplus \mathrm{RPW}^{\mathrm{new}} \oplus h(d_S||\mathrm{ID}_i)$.

(4) Smart card replaces $l', v, c$ with $l'^{\mathrm{new}}, v^{\mathrm{new}}$ and $c^{\mathrm{new}}$.

## 7 Security analysis of Park et al.'s protocol

Part et al. claimed that their protocol is resistant to various known security attacks and can provide user anonymity. However, after a thorough security analysis, we find that the protocol is insecure against user forgery attack, smart card stolen attack and cannot provide user anonymity. In our analysis, we assume that the adversary has all abilities in the security model we defined before.

### 7.1 User forgery attack

By eavesdropping all the messages from the past authentication process, $\mathcal{A}$ can forge a legitimate user and generate session key. The details of this attack can be described as follows.

(1) $\mathcal{A}$ eavesdrops all the authentication messages from the past session and gets $\mathrm{CID}_i$.

(2) $\mathcal{A}$ generates current timestamp $T_1$, random number $r_u$ from $Z_N^*$, computes $R_u = r_u P$, $R = r_u Q_S$, $\mathrm{Auth}_u = h(\mathrm{ID}_i||R||\mathrm{CID}_i||T_1)$ and sends $\{\mathrm{Auth}_u, \mathrm{CID}_i, R_u, T_1\}$ to server $S$.

(3) After receiving $\{\mathrm{Auth}_S, R_S, T_2\}$, $\mathcal{A}$ generates current timestamp $T_3$, computes $\mathrm{SK}_u = r_u R_S$, $\mathrm{SK} = \mathrm{kdf}(\mathrm{ID}_i||\mathrm{SK}_u||T_1||T_2)$, $\mathrm{Auth}_{us} = h(R||\mathrm{SK}||T_3)$, and sends $\{\mathrm{Auth}_{us}, T_3\}$ to $S$.

It is clear if $\mathcal{A}$ captures $\mathrm{CID}_i$ from the past session, he/she can forge a legitimate user.

### 7.2 Lack of user anonymity

If $\mathcal{A}$ can successfully capture all the authentication messages in public network, he/she can easily link messages from different sessions to the same user. The details of this attack can be described as follows.

(1) $\mathcal{A}$ captures all the authentication messages in the public network and gets various $\mathrm{CID}_i$ from the different session.

(2) $\mathcal{A}$ compares these $\mathrm{CID}_i$ and if two sessions have the same $\mathrm{CID}_i$, then $\mathcal{A}$ can link these sessions to the same user.

It is clear that $\mathcal{A}$ can link two transcripts to one user and Park et al.'s protocol cannot provide user anonymity.

### 7.3 Smart card stolen attack

In Part et al.'s protocol, the secret data is stored in the smart card. As we all know, smart card is pretty easy to steal or copy. Assume $\mathcal{A}$ has gain access to user's smart card, he/she can easily perform password guessing attack and forge a legitimate user. The details of this attack can be demonstrated as follows.

(1) $\mathcal{A}$ gets $U_i$'s smart card (or mobile device) and get secret data $\{l', v, c\}$.

(2) $\mathcal{A}$ guesses the possible password $\mathrm{PW}_i^*$ and computes $\mathrm{RPW}^* = h(\mathrm{ID}_i||\mathrm{PW}_i^*)$, $a_i^* = \mathrm{RPW}^* \oplus v$, $c^* = h(\mathrm{ID}_i||\mathrm{PW}_i^*||a_i^*)$. $\mathcal{A}$ compare $c^*$ with $c$. If it is true, $\mathcal{A}$ successfully guesses the password, otherwise, $\mathcal{A}$ continually chooses possible passwords.

(3) $\mathcal{A}$ can forge this user.

It is clear that $\mathcal{A}$ can guess the correct password and forge the legitimate user.

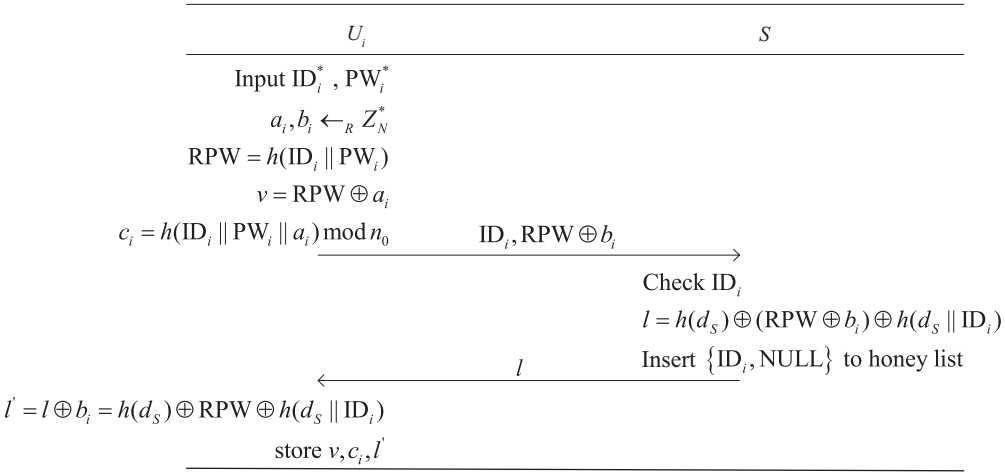


**Figure 3** User registration parse for our protocol.

## 8 Our protocol

In this session, we present our protocol. There are 4 phases in our protocol: (1) system initialization, (2) user registration, (3) user authentication and key agreement, and (4) password change.

### 8.1 System initialization

This phase is conducted by server $S$. It can be demonstrated as follows:

(1) $S$ chooses its $d_S, n_0$ from $Z_N^*$ and uses $d_S$ as its private key.

(2) $S$ chooses a point $P$ from ellipse curve and computes public key $Q_S = d_S P$. Then $S$ regards $\{n_0, P, Q_S, G\}$ as public parameters.

(3) $S$ initial a "Honey List" in which every entry is initialized as {NULL, NULL}. Note that, in our scheme, the server use the "Honey List" to store the clients that have registered. In such a manner, the malicious client cannot register multiple times.

### 8.2 User registration

Before the user can get service from the server, he/she must register him/her self in $S$. As presented in Figure 3, the details of this phase can be demonstrated as follows.

(1) $U_i$ chooses his/her identity $\mathrm{ID}_i$ and password $\mathrm{PW}_i$. Smart card computes $\mathrm{RPW} = h(\mathrm{ID}_i||\mathrm{PW}_i)$, $v = \mathrm{RPW} \oplus a_i$, $c_i = h(\mathrm{ID}_i||\mathrm{PW}_i||a_i) \bmod n_0$, $U_i$ sends $\mathrm{ID}_i, \mathrm{RPW} \oplus b_i$ to $S$ via a secret channel where $a_i, b_i$ are random number chosen from $Z_N^*$.

(2) $S$ checks the existence of $\mathrm{ID}_i$, if it exists, asks user to supply another $\mathrm{ID}_i$. Otherwise, $S$ computes $l = h(d_S) \oplus (\mathrm{RPW} \oplus b_i) \oplus h(d_S||\mathrm{ID}_i)$ and inserts $\{\mathrm{ID}_i, 0\}$ into "Honey List".

(3) $S$ sends $l$ back to $U_i$ via a secret channel.

(4) $U_i$ computes $l' = l \oplus b_i = H(d_S) \oplus \mathrm{RPW} \oplus h(d_s||\mathrm{ID}_i)$ then saves $v, c_i, l'$ in smart card.

### 8.3 User authentication and key agreement

In this phase, $U_i$ and $S$ authenticate each other and generate a session key for future communication. This parse can be graphically presented as Figure 4 and be demonstrated as follows.

(1) $U_i$ inputs his/her $\mathrm{ID}_i^*$ and password $\mathrm{PW}_i^*$, computes $\mathrm{RPW}^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*)$, $a_i^* = v \oplus \mathrm{RPW}^*$, $c_i^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*) \bmod n_0$. Smart card checks the equation $c_i^* = c_i$, if it holds, smart card authenticates $U_i$.

(2) $U_i$ generates current timestamp $T_1$ and chooses random number $r_u$ from $Z_N^*$. $U_i$ computes $R_u = r_u P$, $R = r_u Q_S$, $\mathrm{CID}_i = l' \oplus \mathrm{RPW}^* = H(d_S) \oplus h(d_S \oplus \mathrm{ID}_i)$, $\mathrm{DID}_i = \mathrm{ID}_i^* \oplus H(R)$, $\mathrm{Auth}_u = h(\mathrm{ID}_i^*||R||\mathrm{CID}_i||T_1)$. Then $U_i$ sends $\{\mathrm{Auth}_u, \mathrm{DID}_i, R_u, T_1\}$ to $S$ via a public channel.

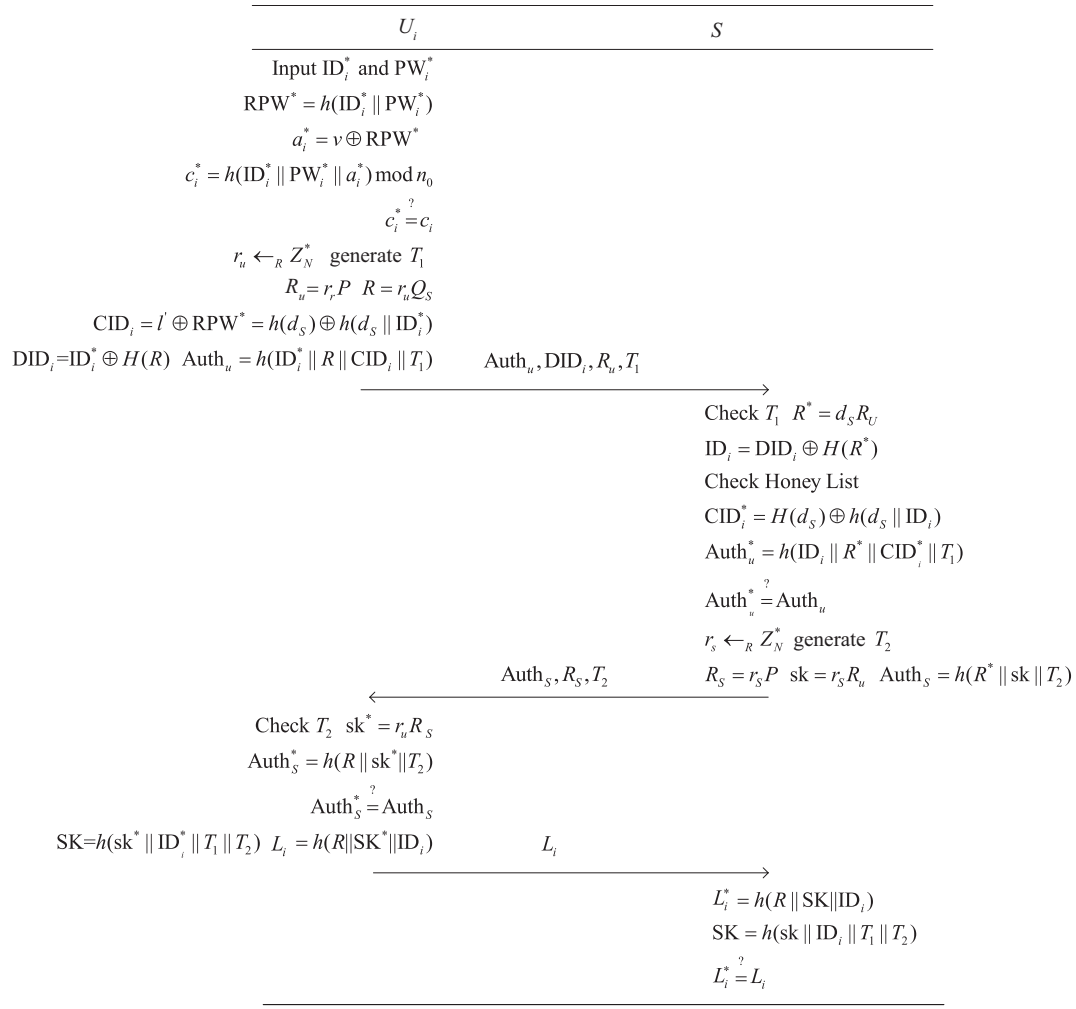


**Figure 4** User authentication and key agreement parse for our protocol.

(3) After receiving messages from $U_i$, $S$ checks timestamp $T_1$, if it is valid, $S$ proceeds the process, otherwise, it rejects the authentication request. Then $S$ computes $R^* = d_S R_u$, $\mathrm{ID}_i = \mathrm{DID}_i \oplus H(R^*)$. $S$ checks the "Honey List" based on $\mathrm{ID}_i$, if the corresponding entry exceeds the predefined threshold, $S$ rejects the request from $U_i$. After that, $S$ computes $\mathrm{CID}_i^* = H(d_S) \oplus h(d_S||\mathrm{ID}_i)$, $\mathrm{Auth}_u^* = h(\mathrm{ID}_i||R^*||\mathrm{CID}_i^*||T_1)$. $S$ checks the equation $\mathrm{Auth}_u = \mathrm{Auth}_u^*$, if it is true, $S$ validates $U_i$.

(4) $S$ chooses random number $r_S$ from $Z_N^*$ and generates current timestamp $T_2$. Then $S$ computes $R_S = r_S P$, $\mathrm{sk} = r_S R_u$, $\mathrm{Auth}_S = h(R^*||\mathrm{sk}||T_2)$ and sends $\{\mathrm{Auth}_S, R_S, T_2\}$ to $U_i$ via a public channel.

(5) $U_i$ receives authenticate messages from $S$ and checks the timestamp $T_2$. if it is valid, $U_i$ proceeds the process. Then, $U_i$ computes $\mathrm{sk}^* = r_u R_S$, $\mathrm{Auth}_S^* = h(R||\mathrm{sk}^*||T_2)$ and checks the equation $\mathrm{Auth}_S = \mathrm{Auth}_S^*$, if it is true, $U_i$ calculates session key $\mathrm{SK} = h(\mathrm{sk}^*||\mathrm{ID}_i||T_1||T_2)$ and $L_i = h(R||\mathrm{SK}||\mathrm{ID}_i)$. $U_i$ sends $\{L_i\}$ to $S$ via a public channel.

(6) $S$ computes $\mathrm{SK} = h(\mathrm{sk}||\mathrm{ID}_i||T_1||T_2)$, $L_i^* = h(R^*||\mathrm{SK}||\mathrm{ID}_i)$ and checks whether $L_i^* = L_i$, if it is true, $S$ stores $\mathrm{SK}^*$ as the session key.

### 8.4 Password change

In this parse, $U_i$ can change his/her password without the assistance of $S$. As presented in Figure 5, the details of this parse can be demonstrated as follows.

(1) $U_i$ inputs his/her identity $\mathrm{ID}_i^*$ and old password $\mathrm{PW}_i^*$, then compute $\mathrm{RPW}^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*)$, $a_i^* = \mathrm{RPW}^* \oplus v$, $c_i^* = h(\mathrm{ID}_i^*||\mathrm{PW}_i^*||a_i^*) \bmod n_0$. If the equation $c_i^* = c_i$ is true, $U_i$ is authenticated by

| $U_i$ |
|---|
| Input $\mathrm{ID}_i^*$ , $\mathrm{PW}_i^*$ |
| $\mathrm{RPW}^* = h(\mathrm{ID}_i^* \,\|\| \,\mathrm{PW}_i^*)$ |
| $a_i^* = v \oplus \mathrm{RPW}^*$ |
| $c_i^* = h(\mathrm{ID}_i^* \,\|\| \,\mathrm{PW}_i^* \,\|\, a_i^*) \bmod n_0$ |
| $c_i^* \stackrel{?}{=} c_i$ |
| input new password $\mathrm{PW}_i^{\mathrm{new}}$ |
| $c_i^{\mathrm{new}} = h(\mathrm{ID}_i^* \,\|\| \,\mathrm{PW}_i^{\mathrm{new}} \,\|\, a_i^*) \bmod n_0$ |
| $l'^{\mathrm{new}} = l' \oplus \mathrm{RPW}^* \oplus \mathrm{RPW}^{\mathrm{new}}$ |

**Figure 5** Password change parse.

the smart card.

(2) $U_i$ inputs his/her new password $\mathrm{PW}_i^{\mathrm{new}}$ and compute $\mathrm{RPW}^{\mathrm{new}} = h(\mathrm{ID}_i^*||\mathrm{PW}_i^{\mathrm{new}})$, $c_i^{\mathrm{new}} = h(\mathrm{ID}_i^*||\mathrm{PW}_i^{\mathrm{new}}||a_i^*) \bmod n_0$, $l'^{\mathrm{new}} = l' \oplus \mathrm{RPW}^* \oplus \mathrm{RPW}^{\mathrm{new}}$. The smart card replaces $\{l', v, c_i\}$ with $\{l'^{\mathrm{new}}, v, c_i^{\mathrm{new}}\}$.

## 9 Security analysis

In this section, we provide a security discussion and a formal security analysis to show that our protocol is provably secure and is resistant to various security attacks.

### 9.1 Security discussion

In this subsection, we prove our proposal can fulfill all the security requirements and can resist various security attacks.

• **Mutual authentication:** With our protocol, users and server can achieve mutual authentication. Firstly, $U_i$ can authenticate server $S$ by checking the authentication message $\mathrm{Auth}_S = h(R||\mathrm{sk}||T_2)$ during the authentication phase. As long as the private key $d_S$ is remained secure, only real server can compute $R = d_S R_u$. Thus, $U_i$ can authenticate the sever. Secondly, $S$ can authenticate $U_i$ by checking $\mathrm{Auth}_u = h(\mathrm{ID}_i||R||\mathrm{CID}_i||T_1)$ since only $U_i$ can compute $\mathrm{CID}_i$ correctly. Until now, we have proved that our protocol can achieve mutual authentication.

• **User anonymity:** With our protocol, any two messages from different sessions cannot be linked to the same user. In our protocol, $U_i$ has different pseudonyms for each session, i.e., $\mathrm{DID}_i = \mathrm{ID}_i||H(R)$, where $R = r_u Q_S$ is different in each session. Thus, $\mathcal{A}$ cannot identify the same user from different sessions.

• **Forward secrecy:** Our protocol achieves the dependence between long term private key and session key. Any adversary $\mathcal{A}$ who has access to $U_i$'s private key is unable to calculate old session keys. We will prove this as follows.

(1) The session key $\mathrm{SK} = h(\mathrm{sk}||\mathrm{ID}_i||T_1||T_2)$ where $\mathrm{sk} = r_S r_u P$.

(2) Cracking $\mathrm{sk} = r_S r_u P$ through $r_u P$ and $r_S P$ means solving the ECDH problem which is considered computationally impossible.

Thus, our protocol can provide forward secrecy.

Furthermore, the proposed protocol can resist against the following attacks.

• **User forgery attack:** With our protocol, $\mathcal{A}$ cannot forge a legitimate user even if $\mathcal{A}$ gets $U_i$'s smart card. We can justify this as follows.

(1) If $\mathcal{A}$ wants to forge $U_i$, then he/she has to generate correct $\{\mathrm{Auth}_u, \mathrm{DID}_i, R_u, T_1\}$ in which $\{\mathrm{DID}_i, R_u, T_1\}$ can be computed by the adversary. This means $\mathcal{A}$ only has to compute $\mathrm{Auth}_u = h(\mathrm{ID}_i||R||\mathrm{CID}_i||T_1)$

(2) In order to calculate $\mathrm{Auth}_u$, $\mathcal{A}$ needs to get correct $\mathrm{CID}_i = l' \oplus \mathrm{RPW} = h(d_S) \oplus h(d_S \oplus \mathrm{ID}_i)$ where $l'$ is stored in the smart card and $\mathrm{RPW} = h(\mathrm{ID}_i||\mathrm{PW}_i)$

(3) Since $\mathcal{A}$ already get $U_i$'s smart card, he/she only has to guess the password correctly. $\mathcal{A}$ can guess the correct password by off-line password guessing attack using the information he/she gets from the smart card.

(4) $\mathcal{A}$ picks a possible password $\mathrm{PW}_i^{\mathrm{guess}}$ and computes $\mathrm{RPW}^{\mathrm{guess}} = h(\mathrm{ID}_i||\mathrm{PW}_i^{\mathrm{guess}})$, $a_i^{\mathrm{guess}} = v \oplus \mathrm{RPW}^{\mathrm{guess}}$, $c_i^{\mathrm{guess}} = h(\mathrm{ID}_i||\mathrm{PW}_i^{\mathrm{guess}}||a_i^{\mathrm{guess}}) \bmod n_0$ and compare $c_i^{\mathrm{guess}}$ with $c_i$ which is stored in the smart card. $\mathcal{A}$ continues guessing until he/she gets a viable password.

(5) $\mathcal{A}$ computes $\mathrm{Auth}_u$ and sends to $S$. However, because of the modular operation in $c_i$, which will fit that equation. Thus, there is a pretty good chance that the adversary will guess a wrong password and miscalculate the $\mathrm{CID}_i$ in $\mathrm{Auth}_u$. Then $S$ will notice the wrong $\mathrm{Auth}_u$ and increase the corresponding entry in "Honey List". Eventually, the value in "Honey List" would expire the predefined threshold and suspend the user.

• **Server forgery attack:** With our protocol, no adversary can forge server. We can justify this as follows.

(1) To successfully forge a server, $\mathcal{A}$ has to generate correct $\mathrm{Auth}_S = h(R||\mathrm{sk}||T_2)$, which means $\mathcal{A}$ has to know $R$ and sk.

(2) Since $R = d_S R_u$ and $\mathrm{sk} = r_S R_u$, $\mathcal{A}$ has to know correct private key of server $d_S$ to computes these data. As long as $d_S$ is secure, $\mathcal{A}$ cannot forge $S$.

• **Smart card stolen attack:** As we mentioned before, even if $\mathcal{A}$ has $U_i$'s smart card, he/she cannot forge a legitimate user. Before $\mathcal{A}$ can guess the correct password, $S$ will terminate the smart card. Thus, our protocol is resistant to smart card stolen attack.

• **Reply attack:** $\mathcal{A}$ cannot forge user or server by replying old messages from past session. Firstly, $U_i$ can detect this attack by checking timestamp $T_2$ or $R$ in $\mathrm{Auth}_S$. Secondly, $S$ can detect this attack by checking timestamp $T_1$ or $R$ in $\mathrm{Auth}_u$. Thus, our protocol can resist reply attack.

• **Privilege insider attack:** In our protocol, instead of sending the password $\mathrm{PW}_i$ to $S$, $U_i$ sends $\mathrm{RPW} \oplus b_i$ to $S$. Therefore $S$ cannot get user's password. Thus, our protocol is resistant to privilege insider attack.

• **Off-line password guessing attack:** In our protocol, even if adversary gets user's smart card, he/she cannot perform password-guessing attack. As we mentioned before, there are various possible passwords that can match the equation $c_i^* = c_i$, which means the only way to confirm the correctness of the guessed password is by sending $\mathrm{Auth}_u$ to $S$. However, we plant a "Honey list" in server which record every fail log attempt. The value in correspond entry will expire the threshold very quickly. Thus, our protocol can prevent password guessing attack.

### 9.2 Formal security analysis

In this subsection, we will prove that our protocol is secure under random oracle (RO) model.

**Theorem 1.** Let $\mathcal{A}$ be the attacker against semantic security of our two-factor authentication protocol $\Pi$ within a time bound $t$ in RO model. Let $C'$, $s'$ are security parameters that follow the Zipf's law [29]. Assume ECCDH assumption holds, then the probability of $\mathcal{A}$ breaking session-key security (SK-security) of $\Pi$ is

$$\begin{aligned}\mathrm{Adv}_{\Pi,D}(\mathcal{A}) \leqslant & \frac{q_h^2}{|\mathrm{Hash}|} + \frac{(q_h+q_e)^2}{|\mathrm{Hash}|} \\ & + \frac{5q_h}{|\mathrm{Hash}|} + 2q_h \mathrm{Adv}_G^{\mathrm{ECCDH}}(\mathcal{A}) \\ & + \frac{2q_h^2}{C' \cdot q_h^{s'}},\end{aligned} \tag{1}$$

where $\mathrm{Adv}_G^{\mathrm{ECCDH}}$ is advantage of $\mathcal{A}$ breaking ECCDH problem, $q_h, q_s$ are number of Hash query and send query, respectively, $q_e$ is the number OS execute query and $|\mathrm{Hash}|$ is the range space of Hash function.

*Proof.* We defines a sequence of six games. For each game $\mathrm{Game}_i$ $(0 \leqslant i \leqslant 5)$, $\mathrm{Succ}_i$ denotes an event wherein $\mathcal{A}$ successfully guesses the bit $c$. The details of each game can be described as follows.

$\text{Game}_0$: This game simulates the real attack performed by $\mathcal{A}$ against the proposed protocol $\Pi$ in RO model. By definition, we have

$$\text{Adv}_{\Pi}^{\text{AKE}} = |2\text{Pr}[\text{Succ}_0] - 1|. \tag{2}$$

$\text{Game}_1$: This game is identical to $\text{Game}_0$ except that it simulates Hash oracle $h$ by maintaining the Hash list HashList with the entry form of {Input, Output}. When answering hash query, if there is an match record for query input, then return correspond Output. Otherwise, oracle generates random value $R \in \{0,1\}$ and returns it. Then, oracle inserts a new entry $\{\text{Input}, R\}$ into HashList. The Excute, Reveal, Send and CorruptSamrtCard oracles are also simulated as polynomial number of queries asked by $\mathcal{A}$. From the perspective of $\mathcal{A}$, this game is no different from the real environment. Thus, we have

$$\text{Pr}[\text{Succ}_1] = \text{Pr}[\text{Succ}_0]. \tag{3}$$

$\text{Game}_2$: This game is perfectly identical to $\text{Game}_1$ except in the situation that the game is terminated when collision occurs in the transcripts $\{\text{Auth}_u, \text{DID}_i, R_u, T_1\}$ and $\{\text{Auth}_S, R_S, T_2\}$. According to birthday paradox, the possibility of collision in Hash query is $\frac{q_h^2}{2|\text{Hash}|}$. Thus, the possibility of collision in our simulation is at most $\frac{(q_h+q_e)^2}{2|\text{Hash}|}$, since $R$ is select randomly. Now, we have

$$|\text{Pr}[\text{Succ}_2] - \text{Pr}[\text{Succ}_1]| \leqslant \frac{q_h^2}{2|\text{Hash}|} + \frac{(q_h+q_e)^2}{2|\text{Hash}|}. \tag{4}$$

$\text{Game}_3$: This game is perfectly identical to $\text{Game}_2$ if the adversary cannot guess the authentication value $\{\text{Auth}_u, \text{Auth}_S\}$ correctly without query oracle. $\text{Game}_2$ and $\text{Game}_3$ are indistinguishable unless user or server instance reject a correct authentication value. Therefore, we have

$$|\text{Pr}[\text{Succ}_3] - \text{Pr}[\text{Succ}_2]| \leqslant \frac{q_h}{|\text{Hash}|}. \tag{5}$$

$\text{Game}_4$: $\text{Game}_4$ is indistinguishable to $\text{Game}_3$ if the adversary cannot guess the correct session key $\text{SK} = h(\text{sk}||\text{ID}_i||T_1||T_2)$ without query Hash oracle. In other word, the calculation of session key is independent of the password and ephemeral keys $r_S r_u P$. $\text{Game}_4$ and $\text{Game}_3$ are prefect identical unless $\mathcal{A}$ query oracle with common value $(\text{sk}||\text{ID}_i||T_1||T_2)$. Thus, $\text{Adv}_G^{\text{ECCDH}}(\mathcal{A}) \leqslant \frac{q_h}{|\text{Hash}|}|\text{Pr}[\text{Succ}_4] - \text{Pr}[\text{Succ}_3]| - \frac{1}{|\text{Hash}|}$. Hence, we have

$$|\text{Pr}[\text{Succ}_4] - \text{Pr}[\text{Succ}_3]| \leqslant q_h \text{Adv}_G^{\text{ECCDH}}(\mathcal{A}) + \frac{q_h}{|\text{Hash}|}. \tag{6}$$

$\text{Game}_5$: This game is identical to $\text{Game}_4$ if the adversary did not make the $(\text{sk}||\text{ID}_i||T_1||T_2)$ Hash oracle in the Test query. The possibility of getting session key with Hash query is less then $\frac{q_h}{2|\text{Hash}|}$, thus we have

$$|\text{Pr}[\text{Succ}_5] - \text{Pr}[\text{Succ}_4]| \leqslant \frac{q_h}{2|\text{Hash}|}. \tag{7}$$

If $\mathcal{A}$ does not query Hash oracle with correct input, then he/she does not have any advantage in distinguishing the real session key and random one. Additionally, the success rate of off-line password guessing attack is $\frac{q_h^2}{C' \cdot q_h^{s'}}$. Combining result from (2)–(7), we have

$$\begin{aligned} \text{Adv}_{\Pi,D}(\mathcal{A}) \leqslant & \frac{q_h^2}{|\text{Hash}|} + \frac{(q_h+q_e)^2}{|\text{Hash}|} \\ & + \frac{5q_h}{|\text{Hash}|} + 2q_h \text{Adv}_G^{\text{ECCDH}}(\mathcal{A}) \\ & + \frac{2q_h^2}{C' \cdot q_h^{s'}}. \end{aligned} \tag{8}$$

## 10 Performance analysis

In this section, we provide performance analysis between our protocol and several other published protocols. And we can prove that our protocol can fulfill more security requirements and has low computational and communication costs.

**Table 2** Security requirement comparison

| SR | Part et al. [12] | Lu et al. [30] | Ours |
|---|---|---|---|
| Smart card stolen attack | N | N | Y |
| Mutual authentication | N | Y | Y |
| User forgery attack | N | Y | Y |
| Server forgery attack | Y | Y | Y |
| Reply attack | Y | Y | Y |
| Insider attack | Y | Y | Y |
| Forward security | Y | Y | Y |
| User anonymity | N | Y | Y |
| Password guessing attack | N | N | Y |
| Correct login and password change phase | Y | Y | Y |

**Table 3** Computational costs comparison

| Phase | Part et al. [12] | Lu et al. [30] | Ours |
|---|---|---|---|
| System initialization phase | $T_{\mathrm{PM}}$ | $T_{\mathrm{PM}}$ | $T_{\mathrm{PM}}$ |
| User registration phase | $6T_H$ | $5T_H$ | $2T_H$ |
| Authentication and key agreement phase | $6T_{\mathrm{PM}} + 11T_H$ | $11T_{\mathrm{PM}} + 15T_H$ | $6T_{\mathrm{PM}} + 12T_H$ |
| Password change phase | $4T_H$ | $3T_H$ | $3T_H$ |
| Total costs | $7T_{\mathrm{PM}} + 21T_H$ | $12T_{\mathrm{PM}} + 23T_H$ | $7T_{\mathrm{PM}} + 23T_H$ |

### 10.1 Security requirement comparison

In this subsection, we provide a security requirement comparison between proposed protocol and other known protocols. From Table 2 [12,30], it is clear that among these protocols, our protocol can withstand more security attacks.

### 10.2 Computational costs

In this subsection, we present a computational costs comparison between our protocol and others. To begin this comparison, we define some notions which may be used.

- $T_H$: The average time required by a secure Hash function.
- $T_{\mathrm{PM}}$: The average time required by a point multiplication.

The comparison result is presented in Table 3 [12,30]. In order to show the result more vividly, we refer some computation result reported in [31], $T_H \approx 0.0001$ ms, $T_{\mathrm{PM}} \approx 0.0442$ ms, we have our computational result $T = 0.3117$ ms.

### 10.3 Communication costs

In this subsection, we will present a communication costs comparison result. In order to perform this comparison, we assume 128-bit for each random nonce, timestamp, password and identity, 268-bit length for Hash digest and 320-bit for ECC point. As shown in Figure 6, it is clear that our protocol has considerably low communication costs.

## 11 Conclusion

The paper proposed a privacy preserving two-factor user authentication protocol for the SPV nodes in the Bitcoin network. We thoroughly analysis Park et al.'s proposal and spot the protocol is vulnerable to user forgery attack, smart card stolen attack and also fail to provide user anonymity. Thus, we design a new

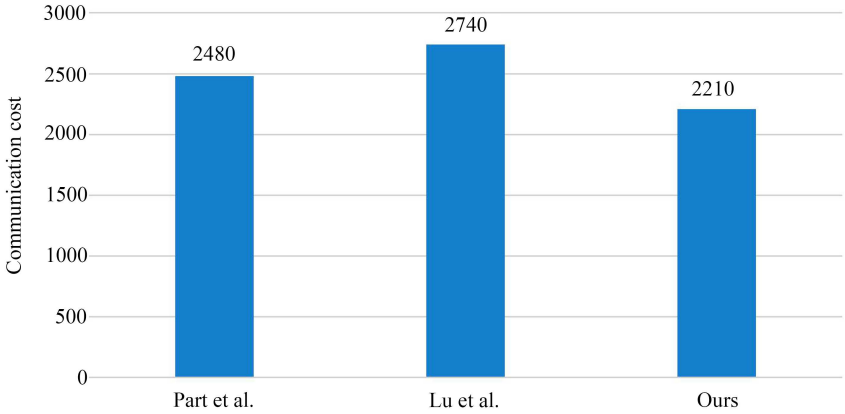


**Figure 6** (Color online) Communication costs comparison.

improved user authentication protocol which is provably secure and can satisfy all security requirements. Additionally, we provide a performance analysis which shows our proposal is practical in the Bitcoin network.

**Acknowledgements** Chunpeng GE was supported by National Natural Science Foundation of China (Grant No. 61702236) and Changzhou Sci & Tech Program (Grant No. CJ20179027). Chunhua SU was supported by JSPS Kiban(B) (Grant No. 18H03240) and JSPS Kiban(C) (Grant No. 18K11298).

**References**

1 Market B. Bitcoin market. 2019. https://coinmarketcap.com/zh/currencies/bitcoin/
2 Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf
3 Wang D, Cheng H B, Wang P, et al. Zipf's law in passwords. IEEE Trans Inform Forensic Secur, 2017, 12: 2776–2791
4 Lamport L. Password authentication with insecure communication. Commun ACM, 1981, 24: 770–772
5 Das M L, Saxena A, Gulati V P. A dynamic ID-based remote user authentication scheme. IEEE Trans Consumer Electron, 2004, 50: 629–631
6 Yoon E-J, Ryu E-K, Yoo K-Y. Further improvement of an efficient password based remote user authentication scheme using smart cards. IEEE Trans Consumer Electron, 2004, 50: 612–614
7 Das M L. Two-factor user authentication in wireless sensor networks. IEEE Trans Wirel Commun, 2009, 8: 1086–1090
8 Khan M K, Alghathbar K. Cryptanalysis and security improvements of 'two-factor user authentication in wireless sensor networks'. Sensors, 2010, 10: 2450–2459
9 Jiang Q, Ma J F, Lu X, et al. An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. Peer-to-Peer Netw Appl, 2015, 8: 1070–1081
10 Wang D, Wang P. Two birds with one stone: two-factor authentication with security beyond conventional bound. IEEE Trans Depend Secure Comput, 2018, 15: 708–722
11 Zhang G M, Yan C, Ji X Y, et al. Dolphinattack: inaudible voice commands. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, 2017. 103–117
12 Park K, Park Y, Park Y, et al. 2PAKEP: provably secure and efficient two-party authenticated key exchange protocol for mobile environment. IEEE Access, 2018, 6: 30225–30241
13 He D B, Chen J H, Hu J. An ID-based client authentication with key agreement protocol for mobile client-server environment on ECC with provable security. Inf Fusion, 2012, 13: 223–230
14 Wu Z Y, Lee Y C, Lai F P, et al. A secure authentication scheme for telecare medicine information systems. J Med Syst, 2012, 36: 1529–1535
15 He D B, Chen J H, Zhang R. A more secure authentication scheme for telecare medicine information systems. J Med Syst, 2012, 36: 1989–1995
16 Wei J H, Hu X X, Liu W F. An improved authentication scheme for telecare medicine information systems. J Med Syst, 2012, 36: 3597–3604
17 Wang D, He D B, Wang P, et al. Anonymous two-factor authentication in distributed systems: certain goals are beyond attainment. IEEE Trans Dependable Secure Comput, 2015, 12: 428–442
18 Tsai J L, Lo N W, Wu T C. Novel anonymous authentication scheme using smart cards. IEEE Trans Ind Inf, 2013, 9: 2004–2013
19 Li C T. A new password authentication and user anonymity scheme based on elliptic curve cryptography and smart card. IET Inform Secur, 2013, 7: 3–10
20 Memon I, Hussain I, Akhtar R, et al. Enhanced privacy and authentication: an efficient and secure anonymous communication for location based service using asymmetric cryptography scheme. Wirel Pers Commun, 2015, 84: 1487–1508

21 Reddy A G, Das A K, Yoon E J, et al. A secure anonymous authentication protocol for mobile services on elliptic curve cryptography. IEEE Access, 2016, 4: 4394–4407
22 Chaudhry S A, Naqvi H, Sher M, et al. An improved and provably secure privacy preserving authentication protocol for SIP. Peer-to-Peer Netw Appl, 2017, 10: 1–15
23 Feng Q, He D B, Zeadally S, et al. Ideal lattice-based anonymous authentication protocol for mobile devices. IEEE Syst J, 2018, 13: 2775–2785
24 Qi M P, Chen J H. An efficient two-party authentication key exchange protocol for mobile environment. Int J Commun Syst, 2017, 30: e3341
25 Wang D, Zhang Z J, Wang P, et al. Targeted online password guessing: an underestimated threat. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016. 1242–1254
26 Chen X F, Li J, Huang X Y, et al. New publicly verifiable databases with efficient updates. IEEE Trans Dependable Secure Comput, 2015, 12: 546–556
27 Zhu Y M, Fu A M, Yu S, et al. New algorithm for secure outsourcing of modular exponentiation with optimal checkability based on single untrusted server. In: Proceedings of 2018 IEEE International Conference on Communications (ICC). New York: IEEE, 2018. 1–6
28 Chen X F, Li J, Huang X Y, et al. Secure outsourced attribute-based signatures. IEEE Trans Parallel Distrib Syst, 2014, 25: 3285–3294
29 Wu F, Xu L L, Kumari S, et al. An improved and provably secure three-factor user authentication scheme for wireless sensor networks. Peer-to-Peer Netw Appl, 2018, 11: 1–20
30 Lu Y R, Li L X, Peng H P, et al. An anonymous two-factor authenticated key agreement scheme for session initiation protocol using elliptic curve cryptography. Multimed Tools Appl, 2017, 76: 1801–1815
31 He D B, Zeadally S, Xu B, et al. An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks. IEEE Trans Inform Forensic Secur, 2015, 10: 2681–2691