Robust Smart Card based Password Authentication Scheme against Smart Card Security Breach

Ding Wang[1,2] and Chun-guang Ma[1]

[1] Harbin Engineering University, Harbin City 150001, China
[2] Automobile Management Institute of PLA, Bengbu City 233011, China
wangdingg@mail.nankai.edu.cn; machunguang@hrbeu.edu.cn

Abstract. As the most prevailing two-factor authentication mechanism, smart card based password authentication has been a subject of intensive research in the past decade and hundreds of this type of schemes have been proposed. However, most of them were found severely flawed, especially prone to the smart card security breach problem, shortly after they were first put forward, no matter the security is heuristically analyzed or formally proved. In SEC'12, Wang pointed out that, the main cause of this issue is attributed to the lack of an appropriate security model to fully identify the practical threats. To address the issue, Wang presented three kinds of security models, namely Type I, II and III, and further proposed four concrete schemes, only two of which, i.e. PSCAV and PSCAb, are claimed to be secure under the harshest model, i.e. Type III security model. However, in this paper, we demonstrate that PSCAV still cannot achieve the claimed security goals and is vulnerable to an offline password guessing attack and other attacks in the Type III security mode, while PSCAb has several practical pitfalls. As our main contribution, a robust scheme is presented to cope with the aforementioned defects and it is proven to be secure in the random oracle model. Moreover, the analysis demonstrates that our scheme meets all the proposed criteria and eliminates several hard security threats that are difficult to be tackled at the same time in previous scholarship, which highly indicates the settlement of an open problem raised by Madhusudhan and Mittal in 2012.

Keywords: Cryptanalysis, Authentication protocol, Smart card, Non-tamper resistant, Dynamic ID, Offline password guessing attack.

1 Introduction

Password authentication with smart card is one of the most convenient and effective two-factor authentication mechanisms for remote systems to assure one communicating party of the legitimacy of the corresponding party by acquisition of corroborative evidence. This technique has been widely deployed for various kinds of authentication applications, such as remote host login, online banking,

e-commerence and e-health [9]. In addition, it constitutes the basis of three-factor authentication [18]. However, there still exists challenges in both security and performance aspects due to the stringent security requirements and resource-strained characteristics of the clients.

Since Chang and Wu [7] introduced the first remote user authentication scheme using smart cards in 1993, there have been many of such schemes proposed [10,11, 20,24,25,27,39,45–48,52]. One prominent issue in this type of schemes is security against offline guessing attack, which is the severest threat that a sound and practical scheme must be able to thwart. Traditionally, to prevent an adversary from launching offline guessing attack, one need to make sure that the scheme is not going to leak any information useful about the client's password to the adversary in the protocol run, even though the password is considered to be weak and low-entropy. By observing this, many schemes employed some techniques similar to Bellovin and Merritt's Encrypted Key Exchange protocol. A common feature of such schemes is that the smart card is assumed to be tamper-resistant, i.e., the secret parameters stored in the smart card cannot be revealed. However, recent research results have demonstrated that the secret data stored in the smart card could be extracted by some means, such as monitoring the power consumption [22, 33] or analyzing the leaked information [19]. Therefore, such schemes [10, 20, 26, 46] based on the tamper resistance assumption of the smart card are vulnerable to offline password guessing attacks, user impersonation attack, etc, once an adversary has obtained the secret data stored in a user's smart card and/or just some intermediate computational results in the smart card. Consequently, a stronger notion of security against offline guessing attack is developed to require that compromising a client's smart card should do not help the adversary launch offline guessing attack against the client's password.

In ICICS'06, seeing that most of the previous schemes have been found inconsistent with this strong notion of security due to the lack of formal security analysis, Yang et al. [52] presented a formal adversarial model for analyzing the security of this type of schemes and a generic construction framework with a new scheme to demonstrate its effectiveness. In 2010, Pu [34] pointed out Yang et al.'s scheme is vulnerable to key compromise attack. Surprisingly, we found Yang et al.'s scheme still cannot achieve its claimed main security goal by demonstrating an offline password guessing attack in Appendix A, and through the security analysis of Yang et al.'s scheme, some subtleties and challenges in designing this type of schemes, different from the traditional password-based authentication, are uncovered. Despite of this, Yang et al.'s formal adversary model does capture the exact two-factor authentication of smart-card-based password authentication schemes: only with both the smart card and the correct password can a user carry out the smart-card-based password authentication scheme successfully with the remote authentication server.

Following Yang et al.'s seminal work [52, 53], many enhanced schemes [17, 24, 25, 35, 38, 39, 42, 44, 45, 48, 50, 51] have been proposed to address the smart card security breach problem, however, most of them were shortly found having various security weaknesses being overlooked [27,36,37,43,48]. Remarkably, some

of them, like [35, 42, 44, 45, 48, 51, 54], even have been provided with a formal proof. The past thirty years of research in the area of password-authenticated key exchange (PAKE) has proved that it is incredibly difficult to get even a single factor based authentication scheme right [5, 16], while the past decade of research in smart card based password authentication has proved that designing a secure and practical two-factor authentication protocol can only be harder.

In SEC'12, Wang [47] observed that the previous papers in this area present attacks on protocols in previous papers and propose new protocols without proper security justification (or even a security model to fully identify the practical threats), which contributes to the main cause of the above failure. Accordingly, Wang presented three kinds of security models, namely Type I, II and III, and further proposed four concrete schemes, only two of which, i.e. PSCAb and PSCAV, are claimed to be secure under the harshest model, i.e. Type III security model. The type III model will be reviewed later in Section 2. However, PSCAb requires Weil or Tate pairing operations to defend against offline guessing attack and may not be suitable for systems where pairing operations are considered to be too expensive or infeasible to implement. Moreover, PSCAb suffers from the well-known key escrow problem and lacks some desirable features such as local password update, repairability and user anonymity. As for PSCAV, in Appendix B, we will demonstrate that it still cannot achieve the claimed security goals and is vulnerable to an offline password guessing attack and other attacks under the Type III security model. As our main contribution, a robust and efficient scheme is presented to cope with the identified defects and it is formally proven to be secure in the Type III security model.

The remainder of this paper is organized as follows: in Section 2, we elaborate on the adversary model and evaluation criteria. Our proposed scheme is presented in Section 3, and its security analysis is given in Section 4. The comparison of the performance of our scheme with the other related schemes is shown in Section 5. Section 6 concludes the paper. An offline password guessing attack against the scheme proposed by Yang et al. [52] is uncovered in Appendix A. The demonstration of offline password guessing attack on Wang's PSCAV protocol, is given in Appendix B.

2 Adversary model and evaluation criteria

There have been several papers dealing with smart-card-based password authentication schemes in recent years (see, e.g., [11, 17, 20, 26, 42, 46, 50]). However, in most of these studies, the authors present attacks on previous schemes and propose new protocols with assertions of the superior aspects of their schemes, while ignoring benefits that their scheme doesn't attempt (or fail) to provide, thus overlooking dimensions on which it fares poorly. Despite the lack of evaluation criteria, another common feature of these studies is that, there is no proper security justification (or even an explicit security model) presented. Accordingly, in the following, an adversary model consistent with the reality is explicitly defined and a comprehensive criteria set is proposed.

2.1 Adversary model

In the conventional password authenticated key exchange (PAKE) protocols, the attacker is modeled to have the full control of the communication channel between the communicating parties [3], such as eavesdropping, intercepting, inserting, deleting, and modifying any transmitted messages over the public channel. Though this assumption is reasonable for password-based authentication situations, it is not sufficient for password-based remote authentication using smart cards.

Recent studies have shown that the secret data in the smart card could be extracted out by monitoring power consumption or employing reverse engineering techniques [19, 22, 30]. Consequently, the leakage of sensitive parameters stored in the smart card may lead the original secure schemes vulnerable to smart card loss problem, such as offline password guessing attack and impersonation attack [24, 27, 39]. Secondly, as observed and in-deep investigated by Wang [47] quite recently, malicious card readers also contribute to the security failures of such schemes. Once the card reader is under the control of the attacker, the smart card owner's input password may be intercepted (but the secret information stored in the card should not be revealed at the same time, the simple reason is that an attacker cannot read the sensitive information on the card within a short time period). However, we restrict the attacker from first intercepting the password via the card reader and then reading the information stored in the card via the stolen smart card, otherwise the security failure is unavoidable.

In reality, previous session key(s) may be lost for a variety of reasons, including hacking, mishandling of information and the prescribed-release of that session key when the session is torn down. Adding this ability to $\mathcal{A}$ allows our model capture the threat of the known key attack. To evaluate the damage of leakage of server's long-term private key, the capability of learning server's long-time private key is equipped with our adversary, including it allows us to deal with forward secrecy and key compromise impersonation attack.

Furthermore, it is worth noting that, in remote user authentication schemes, for the sake of user-friendliness, a user is often allowed to select her own identity ID at will (at most confined to a predefined format) during the registration phase; the user usually tends to chose an identity which is easily remembered for her convenience. Consequently, these easy-to-remember identities are of low entropy and thus can also be offline enumerated by an adversary $\mathcal{A}$ within polynomial time in the same way with the passwords. Hence, in practice, it is reasonable and realistic to assume that $\mathcal{A}$ can offline enumerated all the (ID, PW) pairs in the Cartesian product $\mathcal{D}_{id} * \mathcal{D}_{pw}$ within polynomial time. In contrast, most of the proposed dynamic-ID schemes (i.e. user's identity is concealed in session-variant pseudo-identities to provide the property of user anonymity), e.g. [27, 39, 43, 44], explicitly assume $\mathcal{A}$ cannot guess both ID and PW correctly at the same time. In other words, such dynamic-ID schemes may be vulnerable to offline password guessing attack under our assumption.

The capabilities of the adversary in our model are summarized in Table 1. As far as we know, our work, following Li-Lee's [24] and Wang's [47] work, is one

Table 1. Capabilities of the adversary

C-0	The adversary $\mathcal{A}$ can enumerate offline all the elements in the Cartesian product $\mathcal{D}_{id} \times \mathcal{D}_{pw}$ within a reasonable amount of time, where $\mathcal{D}_{pw}$ and $\mathcal{D}_{id}$ denote the password space and the identity space, respectively.
C-1	The adversary $\mathcal{A}$ has full control of the communication channel between the communicating parties, such as eavesdropping, intercepting, inserting, deleting, and modifying any transmitted messages over the public channel.
C-2	The adversary $\mathcal{A}$ may either learn (i) the password of a legitimate user via malicious card reader, or (ii) extract the secret information of the smart card by side-channel attacks, but cannot achieve both.
C-3	The adversary $\mathcal{A}$ can learn the previous session key(s).
C-4	The adversary $\mathcal{A}$ has the capability of learning server's long-time private key(s) only when evaluating the eventual failure of the server.

of the few ones that explicitly specify the capabilities of the adversary. In [47], Wang presented three kinds of security models, namely Type I, II and III, and further proposed four concrete schemes, only two of which are claimed to be secure under the harshest model, i.e. Type III security model. In Wang's Type III model, three assumptions are made: 1) an adversary $\mathcal{A}$ is allowed to have full control of the communication channel between the user and the server, which is consistent with C-1; 2) the smart card is assumed to be non-tamper resistant and user's password may be intercepted by $\mathcal{A}$ by malicious card reader, but not both, which is consistent with C-2; 3) there is no counter protection in the smart card, i.e. $\mathcal{A}$ can issue a large amount of queries to the smart card using a malicious card reader to learn some useful information. With regard to assumption 3, we argue that this assumption may not be of much practical significance, because whether assumption 3 is valid or not in practice has little relevance with protocol security under assumption 2. On the one hand, if there is no verification of the input password before the function (the run mode) of the smart card, the only way that $\mathcal{A}$ can learn some useful information (except the static data stored in the card, which can be learned by $\mathcal{A}$ under assumption 2) is to interact with the remote server, which can be effectively thwarted by the server, e.g., locking the corresponding user account after a few failed login attempts. On the other hand, if this verification exists, $\mathcal{A}$ can always find the password that passes the verification by exhaustively inputting her guessing passwords into the malicious card reader(and with assumption 2, secret data stored in the card can also be extracted out), which is explicitly not allowed in the Type III model. Hence, assumption 3 is not incorporated into our model. As with some other studies [39, 43], we may simply assume that there is counter protection in the smart card, i.e., the smart card will be self-destroyed or locked for a time period if the query number exceeds a certain threshold (e.g., the GSM SIM card V2 or later has this capability).

According to the above analysis, our model is in much similarity with the Type III model (the most powerful one) introduced in [47], and the major difference

is that $\mathcal{A}$ in Type III model is not provided with the capabilities of C-3 and C-4, which may lead to the failure of dealing with some important security features, such as forward secrecy and resistance to known key attack. As compared to Li-Lee's adversary model [24] and Yang et al.'s model [52, 53], our model has explicitly taken the malicious card reader into consideration, and $\mathcal{A}$ is further equipped with the capabilities of C-3 and C-4. Moreover, $\mathcal{A}$ in our model is assumed to be able to offline enumerated all the (ID, PW) pairs in the Cartesian product $\mathcal{D}_{id} * \mathcal{D}_{pw}$ within polynomial time, which enables our model to deal with the special security issues, such as resistance to offline password (more precisely, (ID, PW) pair) guessing attack and undetectable online password guessing attack, in dynamic-ID schemes. Consequently, our model is stronger as it incorporates the previous assumptions as well as other new practical assumptions.

2.2 Evaluation criteria

As pointed out in [52], although the construction and security analysis of password-based authentication schemes with smart cards have a long history, there is no common set of desirable security properties that has been widely adopted for the construction of this type of schemes. In 2006, Liao et al. [26] made an attempt to consolidate a large set of ten desirable properties, including six security requirements, for evaluating the goodness of a password-based authentication scheme using smart card. Later on, Yang et al. [52] argued that Liao et al.'s criteria set has some redundancies and proposed a new set of only five criteria for rating the schemes. Yang et al.'s criterion set is too conceptual (and thus ambiguous, not specific) to be adopted in real applications. Almost at the same time, Tsai et al. [41] also presented another list of nine security requirements and ten desirable features that an ideal password authentication scheme should achieve. A common feature of both Liao et al.'s and Tsai et al.'s criteria is that, the security requirements are based on the temper-resistance assumption of the smart cards, which may be inconsistent with the reality when taking into account the state-of-the-art techniques of side-channel cryptanalysis.

More recently, Madhusudhan and Mittal [29] pointed out that earlier criteria sets have redundancies and ambiguities and also proposed a new criteria set of nine security requirements and ten desirable features to evaluate this type of schemes. Since the security requirements of their criteria are based the non-temper resistance assumption of the smart cards, their criteria set is superior to other proposed sets. However, it fails to include some important security requirements for an authentication protocol with key agreement, i.e., resistance to known key attack, key compromise impersonation attack and unknown key share attack [3, 23].

By summarizing these earlier studies, we put forward a comprehensive list of twelve independent criteria in terms of user friendliness and security that a password-based remote user authentication scheme with smart card should satisfy:

C1. the server needs not to maintain a database for storing the passwords or some derived values of the passwords of its clients [26, 29, 41, 53];

C2. the password is memorable, and can be chosen freely and changed locally by the user [26, 29, 41, 53];
C3. the password cannot be derived by the privileged administrator of the server [26, 29, 41, 53];
C4. the scheme is free from smart card loss attack, i.e., unauthorized users should not be able to easily change the password of the smart card, guess the password of the user by using password guessing attacks, or impersonate the user to login to the system, even if the smart card is obtained and/or secret data in the smart card is revealed [**?**, 27, 29, 39, 41, 45, 55];
C5. the scheme can resist various kinds of sophisticated attacks, such as offline password guessing attack, replay attack, parallel session attack, denial of service attack, stolen verifier attack, impersonation attack, key compromise impersonation attack, known key attack [25, 27, 29, 41, 45];
C6. the scheme provides smart card revocation with good repairability, i.e., the client can revoke the smart card without changing her identity [12, 25, 29, 44, 45];;
C7. the client and the server can establish a common session key during the authentication process [25, 27, 29, 39, 41];
C8. the scheme is not prone to the problems of clock synchronization and time-delay [8, 24, 44, 45, 48];
C9. the scheme provides the property of timely wrong password detection, i.e. the user will be timely notified if he inputs wrong password by mistake in login phase [29, 41];
C10. the scheme can achieve mutual authentication [26, 29, 41, 53];
C11. the scheme preserves user anonymity[3] to avoid partial information leakage. [27, 29, 39, 41];
C12. the scheme provides the property of forward secrecy [12, 28, 29, 41, 48].

Our criteria set is a refinement and an extension of some previously proposed requirement sets, it not only eliminates the redundancies and ambiguities of the old requirement sets, but also facilitates cryptanalysis due to its concreteness. It is not difficult to check that Madhusudhan and Mittal's criteria set is entirely included into our set. And it is also worth noting that, unlike the criteria sets proposed by Tsai et al. [41] and Liao et al. [26], the criterion concerning with performance, which says "The scheme must be efficient and practical", is not incorporated into our set. The main reason is that, it does not seem to be measurable without referring to other related schemes, in other words, isolating it from the criteria set can make our set more concrete and decidable. Furthermore, the efficiency of a scheme may depend on the implementation environment, while practicality is largely related to the target applications [53]. Except this criterion, all the other criteria in [41] and [26] are included into our set. In conclusion, our criteria set is more comprehensive and concrete.

[3] Note that, in the scenario of remote user authentication, user anonymity is defined against the public rather than the server because the server has to first identify the legitimacy of the user and then obtain the user's real identity for accounting and billing purposes. [31]

3 Our proposed scheme

In this section, we present a robust and efficient smart card based password authentication scheme that provides all of the twelve criteria introduced in Section 2.2. Our scheme (summarized in Fig.1, in Appendix C) is based on one of our previous works [43] and consists of four phases: the registration phase, the login phase, the verification phase and the password change phase. For ease of presentation, we employ some intuitive abbreviations and notations listed in Table 2.

Table 2. Notations

Symbol	Description
U_i	i^{th} user
S	remote server
$\mathcal{A}$	malicious attacker
SC	smart card
ID_i	identity of user U_i
PW_i	password of user U_i
x	the secret key of remote server S
$\oplus$	the bitwise XOR operation
$\|$	the string concatenation operation
$A \rightarrow B : C$	message C is transferred through a common channel from A to B
$A \Rightarrow B : C$	message C is transferred through a secure channel from A to B

3.1 Registration phase

The protocol is defined over a finite cyclic group $\mathbb{G} =< g >$ of order a ℓ-bit prime number q. This group could be $\mathbb{G} = Z_q^*$, where $Z_q^* = 1, 2, \cdots q-1$, or it could be a subgroup of $GF(p)$, or it could be an elliptic curve group. In this paper, we assume $\mathbb{G}$ is a prime order subgroup of F_p, where p also is a large prime number such that $q|p-1$. Hash functions from $\{0,1\}^* \rightarrow \{0,1\}^{l_i}$ are denoted by $\mathcal{H}_i(i = 0, 1, 2, 3)$, where l_i is the bit length of function output, e.g. $l_i = 2^{160}$. We also define a medium integer n, $2^8 \leq n \leq 2^{16}$, which determines the capacity of the pool of the (ID, PW) pair against offline guessing attack. Let $(x, y = g^x \bmod p)$ denote the server S's private key and its corresponding public key, where x is kept secret by S and y is stored inside each user's smart card. The registration phase involves the following operations:

Step R1. U_i chooses her identity ID_i, password PW_i and a random number b.
Step R2. $U_i \Rightarrow S : \{ID_i, \mathcal{H}_0(b \parallel PW_i)\}$.
Step R3. On receiving the registration message from U_i at time T, S first checks whether U_i, is a registered user. If it is U_i's initial registration, S creates an entry for U_i in the account-database and stores $(ID_i, T_{reg} = T)$ in this entry.[2] Otherwise, S updates the value of T_{reg} with T in the

[2] For the mere sake of achieving provable security, in Section IV we assume another parameter $P_i = \mathcal{H}_0(b\|PW_i)$ is also stored in this entry.

existing entry for U_i. Next, S computes $N_i = \mathcal{H}_0(b \parallel PW_i) \oplus \mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ and $A_i = \mathcal{H}_0((\mathcal{H}_0(ID_i) \oplus \mathcal{H}_0(b \parallel PW_i)) \bmod n)$.

Step R4. $S \Rightarrow U_i$: A smart card containing security parameters $\{N_i, A_i, q, g y, n, \mathcal{H}_0(\cdot), \mathcal{H}_1(\cdot), \mathcal{H}_2(\cdot), \mathcal{H}_3(\cdot)\}$.

Step R5. Upon receiving the smart card SC, U_i enters b into SC.

3.2 Login phase

When U_i wants to login to the system, the following operations will be performed:

Step L1. U_i inserts her smart card into the card reader and inputs ID_i^*, PW_i^*.

Step L2. SC computes $A_i^* = \mathcal{H}_0((\mathcal{H}_0(ID_i^*) \oplus \mathcal{H}_0(b \| PW_i^*)) \bmod n)$ and verifies the validity of ID_i^* and PW_i^* by checking whether A_i^* equals the stored A_i. If the verification holds, it implies $ID_i^* = ID_i$ and $PW_i^* = PW_i$ with a probability of $\frac{n-1}{n} (\approx \frac{99.90}{100}$, when $n = 2^{10})$. Otherwise, the session is terminated.

Step L3. SC chooses a random number u and computes $C_1 = g^u \bmod p$, $Y_1 = y^u \bmod p$, $CID_i = ID_i \oplus \mathcal{H}_0(C_1 \parallel Y_1)$.

Step L4. $U_i \rightarrow S : \{C_1, CID_i\}$.

3.3 Verification phase

After receiving the login request, the server S performs the following operations:

Step V1. S computes $Y_2 = (C_1)^x \bmod p$ using its private key x, and derives $ID_i = CID_i \oplus \mathcal{H}_0(C_1 \parallel Y_2)$ and checks whether ID_i is in the correct format. If ID_i is not valid, the session is terminated. Otherwise, S generates a random number v and computes the temporary key $K_S = (C_1)^v \bmod p$, $C_2 = g^v \bmod p$, $k = \mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ and $C_3 = \mathcal{H}_1(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_S)$, where T_{reg} is extracted from the entry corresponding to ID_i.

Step V2. $S \rightarrow U_i : \{C_2, C_3\}$.

Step V3. On receiving the reply message from the server S, SC computes $k = \mathcal{H}_0(x \parallel ID_i \parallel T_{reg}) = N_i \oplus \mathcal{H}_0(b \parallel PW_i)$, $K_U = (C_2)^u \bmod p$, $C_3^* = \mathcal{H}_1(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_U)$, and compares C_3^* with the received C_3. This equivalency authenticates the legitimacy of the server S, and U_i goes on to compute $C_4 = \mathcal{H}_2(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_U)$.

Step V4. $U_i \rightarrow S : \{C_4\}$

Step V5. Upon receiving $\{C_4\}$ from U_i, the server S first computes $C_4^* = \mathcal{H}_2(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_S)$ and then checks if C_4^* equals the received value of C_4. If this verification holds, the server S authenticates the user U_i and the login request is accepted else the connection is terminated.

Step V6. The user U_i and the server S agree on the common session key $sk_U = \mathcal{H}_3(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_U) = \mathcal{H}_3(ID_i \| lID_S \| Y_1 \| C_2 \| k \| K_S) = sk_S$ for securing future data communications.

3.4 Password change phase

For the sake of security, user friendliness and communication efficiency, this phase is performed locally without the hassle of interaction with the remote authentication server, and it involves the following steps:

Step P1. U_i inserts her smart card into the card reader and inputs ID_i and the original password PW_i.

Step P2. The smart card computes $A_i^* = \mathcal{H}_0((\mathcal{H}_0(ID_i) \oplus \mathcal{H}_0(b \parallel PW_i)) \bmod n)$ and verifies the validity of A_i^* by checking whether A_i^* equals to the stored A_i. If the verification holds, it implies the input ID_i and PW_i are valid. Otherwise, the smart card rejects.

Step P3. The smart card asks the cardholder to resubmit a new password PW_i^{new} and computes $N_i^{new} = N_i \oplus \mathcal{H}_0(b \parallel PW_i) \oplus \mathcal{H}_0(b \parallel PW_i^{new})$, $A_i^{new} = \mathcal{H}_0((\mathcal{H}_0(ID_i) \oplus \mathcal{H}_0(b \parallel PW_i^{new}) \bmod n)$. Then, smart card updates the values of N_i and A_i stored in its memory with N_i^{new} and A_i^{new}, respectively.

Notes and Rationales. To achieve C2 and C4 at the same time, a verification of the authenticity of the original password before updating the value of N_i in the memory of smart card is essential. And thus, besides N_i, some additional parameter(s) should be stored in the smart card, which may introduce new vulnerabilities, such as offline guessing attack and user impersonation attack. To gain a better insight into this issue, now let's assume an additional parameter $A_i = \mathcal{H}_0(ID_i \parallel \mathcal{H}_0(PW_i))$ is stored in the smart card. Whenever U_i wants to change her password, first she must submit her identity ID_i^* and password PW_i^*, then the smart card checks whether $\mathcal{H}_0(ID_i^* \parallel \mathcal{H}_0(PW_i^*))$ equals the stored A_i. One can easily find that an adversary $\mathcal{A}$ can exhaustively search the correct (ID_i, PW_i) pair in an offline manner once the parameter A_i is obtained, which definitely leads to an offline guessing attack, resulting in the violation of C4. What we have just described directly applies to one of our earlier works [43], the parameter A_i in [43] is exactly computed in this insecure manner and thus $\mathcal{A}$ can obtain the correct (ID_i, PW_i) pair once the parameter A_i is revealed under our adversary model illustrated in Table 1 [3].

However, if the parameter A_i is computed as $A_i = \mathcal{H}_0(\mathcal{H}_0(ID_i) \oplus \mathcal{H}_0(PW_i)) \bmod n)$, one can be assured that there exists $\frac{|\mathcal{D}_{id}|*|\mathcal{D}_{pw}|}{n} \approx 2^{30}$ candidates of (ID, PW) pair to frustrate $\mathcal{A}$ when $|\mathcal{D}_{id}| = |\mathcal{D}_{pw}| = 10^6$ [4, 21, 49] and $n = 2^{10}$, where $|\mathcal{D}_{id}|$ and $|\mathcal{D}_{pw}|$ denote the size of the identity space and password space, respectively. In this manner, we prevent $\mathcal{A}$ from obtaining the exactly correct (ID, PW) pair and we call the parameter A_i calculated through this new method "a fuzzy verifier". An obvious "side effect" of this "fuzzy verifier" is that it

[3] In [43], the authors claimed that their scheme can withstand offline guessing attack even if the smart card security is breached. Why we get quite contradictory results? The reason is that the claim made in [43] is based on the assumption that $\mathcal{A}$ cannot guess both ID_i and PW_i correctly at the same time in polynomial time. In other words, $\mathcal{A}$ in [43] is not equipped with the capability C-0 in Table 1. This explicates the paradox.

can be used to fulfill criterion C9. One may argue that what if U_i happens to submit a wrong (ID_i^*, PW_i^*) pair such that $\mathcal{H}_0((\mathcal{H}_0(ID_i^*) \oplus \mathcal{H}_0(PW_i^*)) \bmod n) = A_i$, while $(ID_i^*, PW_i^*) \neq (ID_i, PW_i)$? The reality is that, this possibility is $\frac{1}{n} < \frac{1}{1000}$, which is too small to significantly degrade the effectiveness of C9, and we conjecture that there is an unavoidable trade-off when fulfilling C4 and C9.

To achieve C3, $\mathcal{H}_0(b \parallel PW_i)$ instead of PW_i or $h(PW_i)$ is submitted to server S, where b is a random number unknown to the server S; to achieve C6, an entry (ID_i, T_{reg}) corresponding to U_i is stored in S's database,[4] only T_{reg} needs to be updated when user U_i revokes her smart card; to achieve C8, a nonce based mechanism instead of the timestamp based design is preferred to provide the freshness of the messages; to achieve user anonymity (C11), user's real identity ID_i is concealed in the session-variant pseudo-identity CID_i, by employing an analogous Elgamal encryption; to achieve forward secrecy (C12), Diffie-Hellman key exchange technique is adopted.

4 Security analysis

In the following, we first describe a formal security model for smart card based password authentication schemes, and then show that our scheme is secure in this model under the assumptions that the hash function closely behaves like a random oracle and that the computational Diffie-Hellman problem is difficult. In particular, our protocol achieves forward secrecy property and security against known key attack, key compromise impersonation attack.

5 Security analysis

In the following, we first describe a formal security model for smart card based password authentication schemes, and then show that our scheme is secure in this model under the assumptions that the hash function closely behaves like a random oracle and that the computational Diffie-Hellman problem is difficult. In particular, our protocol achieves forward secrecy property and security against known key attack, key compromise impersonation attack.

5.1 Formal security model

We define some notions and recall the BPR2000 security model [2] where the adversary's capabilities are modelled through queries. However, we do not use the original model directly, but adopt the reified version proposed by Bresson et al. [6] with a few changes so that we can define the special security requirements for password authentication schemes using smart cards. We refer the reader to the original papers for more details.

[4] Note that the storage of such data in server's database will not violate criterion C1.

Players. We denote a server S and a user, or client, U that can participate in the authentication protocol $\mathcal{P}$. Each of them may have several instances called oracles involved in distinct, possibly concurrent, executions of $\mathcal{P}$. We denote client instances and server instances by U^i and $S^j, i, j \in \mathbb{Z}$, and denote any kind of instance by I.

According to our scheme, the server has a long-term private/public key pair $(x, y = g^x)$. User U_i holds a password PW_i, which is uniformly drawn from a small dictionary $\mathcal{D}$ of size $|\mathcal{D}|$. Additionally, when the user U_i enrolls in the server S, S stores N_i and A_i into a smart card and issues it to the user U_i, where N_i and A_i are (injective) transformations of PW_i and x. The assumption of the uniform distribution for the password is just to make notations simpler, but everything would work with any other distribution, replacing the probability $\frac{\lambda}{|\mathcal{D}|}$ by the sum of the probabilities of the λ most probable passwords.

Queries. The interaction between an adversary $\mathcal{A}$ and the protocol participants occurs only via oracle queries, which model the adversary capabilities in a real attack. The query types available to $\mathcal{A}$ are defined as follows.

- Execute(U^i, S^j): This oracle query is used to model passive (eavesdropping) attacks of the adversary. The output of this query consists of the messages that were exchanged during the honest execution of the protocol.
- Send(I, m): This query models an active attack, in which the adversary $\mathcal{A}$ may send a message to instance I and get back the response I generates in processing the message m according to the protocol $\mathcal{P}$. A query Send(U^i, Start) initializes the key exchange protocol. Start is a message, and thus the adversary receives the flow the client should send out to the server.
- Test(I): This oracle query is not used to simulate the adversary's attack, but to define session key's semantic security. If no session key for instance I is defined, then undefined symbol $\perp$ is returned. Otherwise, a private coin c is flipped. If $c = 1$ then the session key sk is returned to $\mathcal{A}$, otherwise a random key of the same size is returned. This query can be called only once during its execution.
- Reveal(I): This query models the misuse of session keys (i.e. C-3 in Table 1). It returns to the adversary the session key sk of participant instance I, if the targeted instance actually "holds" a session key, and I and its partner were not asked by a Test query. Otherwise the $\perp$ is returned.
- Corrupt(I, a): This query models corruption capability of the adversary (i.e. C-2 and C-4 in Table 1). $\mathcal{A}$ can indeed steal/break either one of the two authentication factors of clients, but not both:
  - If $a = 1$, it outputs the password PW_i of U.
  - If $a = 2$, it outputs parameters, i.e. $\{N_i, A_i, b\}$, stored in the smart card.
  - If $a = 3$, it outputs the private key x of S.

It is easy to see that, the above oracle queries indeed can model all the adversary capabilities listed in Table 1.

Partnering. We define partnering by using the notion of session identifier *sid*. Let U^i and S^j be a pair of instances. We say that the instances U^i and S^j

are partnered if the following conditions are satisfied: ① Both instances have accepted; ② Both instances shared the same *sid*; ③The partner identifier (pid) of U^i is S and vice-versa. In general, we let *sid* be the ordered concatenation of all messages sent and received by the instance U^i (or S^j).

Freshness. The freshness notion captures the intuitive fact that a session key can not be trivially known to the adversary. We say that an instance I is fresh if: ① I has accepted and computed a session key; ② Neither I nor its partner have been asked for a Reveal-query; ③ At most one Corrupt-query is made to the client involved (either I or its partner) and no Corrupt-query is made to the server involved (either I or its partner), since the beginning of the game.

Correctness. If U^i and S^j are partnered and they are accepted, then they end up with the same session key $sk_U^i = sk_S^j$.

Authentication. A fundamental goal of the authentication schemes is to prevent the adversary from impersonating the client or the server. We denote by $\mathrm{Adv}_{\mathcal{P}}^{\mathrm{auth}}(\mathcal{A})$ the probability that $\mathcal{A}$ successfully impersonates a client in an execution of $\mathcal{P}$ as an instance of either U or S in an execution of $\mathcal{P}$, which means that S (resp. U) agrees on a key, while the latter is shared with no instance of U (resp. S).

Semantic security. Another major concern of authentication schemes with key agreement is to protect the privacy of the session key. In a protocol execution of $\mathcal{P}$, an adversary $\mathcal{A}$can ask a polynomial number of Execute-query, Reveal-query and Send-query. It can also ask a single Test query to a fresh instance. In the end of the game, $\mathcal{A}$ outputs a guess bit c' for the bit c involved in the Test-query. We say that $\mathcal{A}$ wins the game if $c' = c$, and this event is denoted by Succ. Accordingly, the advantage of $\mathcal{A}$ in breaking the semantic security of the protocol $\mathcal{P}$ is defined to be

$$\mathrm{Adv}_{\mathcal{P}}^{\mathrm{ake}}(\mathcal{A}) = 2\mathrm{Pr}[\mathrm{Succ}(\mathcal{A})] - 1 = 2\mathrm{Pr}[\mathrm{c}' = \mathrm{c}] - 1$$

where the probability space is over all the random coins of the adversary and all the oracles. The protocol $\mathcal{P}$ is said to be semantically secure if any polynomial time (PPT) adversary $\mathcal{A}$'s advantage is negligible in the security parameter l.

5.2 Formal security proof

Before stating the security results, we recall the computational assumption on which the formal security proof relies.

Computational Diffie–Hellman (CDH) Assumption. Let $\mathbb{G}$ be a finite cyclic group of prime order q generated by an element g, where the operation is denoted multiplicatively. A (t, ϵ)-CDH attacker in $\mathbb{G}$ is a probabilistic machine Δ running in time t such that

$$\mathrm{Adv}_{g,\mathbb{G}}^{\mathrm{CDH}}(\Delta) = \Pr_{x,y}[\Delta(g^x, g^y) = g^{xy}] \geq \epsilon,$$
$$\mathrm{Adv}_{g,\mathbb{G}}^{\mathrm{CDH}}(t) = \max_{\Delta}\{\mathrm{Adv}_{g,\mathbb{G}}^{\mathrm{CDH}}(\Delta)\},$$

where the probability is taken over the random values x and y. The CDH-Assumption states that $\mathrm{Adv}^{\mathrm{CDH}}_{g,\mathbb{G}}(t) \leq \epsilon$ for any t/ϵ not too large.

Theorem 1. *Let $\mathbb{G}$ be a representative group and let $\mathcal{D}$ be a uniformly distributed dictionary of size $|\mathcal{D}|$. Let $\mathcal{P}$ be the improved proposed authentication scheme stated in Section 3. Let $\mathcal{A}$ be an adversary against the semantic security within a time bound t, with less than q_{send} Send-queries and q_{exe} Execution-queries, and making less than q_h random oracle queries. Then we have*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{ake}}_{\mathcal{P}}(\mathcal{A}) &= 2\mathrm{Pr}[\mathrm{Succ}_7] - 1 + 2(\mathrm{Pr}[\mathrm{Succ}_7] - \mathrm{Pr}[\mathrm{Succ}_0]) \\ &\leq \tfrac{2q_{send}}{|\mathcal{D}|} + 8q_h\mathrm{Adv}^{\mathrm{CDH}}_{\mathcal{P}}(t + (q_{send} + q_{exe} + 1)\cdot\tau_e) \\ &\quad + \tfrac{q_h^2+6q_{send}}{2^l} + \tfrac{(q_{send}+q_{exe})^2}{p},\end{aligned}$$

where $t' \leq t + (q_s + q_p + 1)\tau_e$, τ_e is the computational time for an exponentiation in $\mathbb{G}$, and $l = min\{l_i\}, i = 0, 1, 2, 3$.

Proof. Let be $\mathcal{A}$ an adversary against the semantic security of our scheme. The idea is to employ $\mathcal{A}$ to construct adversaries for each of the underlying primitives in such a way that if $\mathcal{A}$ succeeds in breaking the semantic security, then at least one of these adversaries manages to breach the security of an underlying primitive. We prove Theorem 1 through a series of hybrid games, starting with the real attack and ending up with a game where $\mathcal{A}$'s advantage is 0, and for which we can bound the difference in $\mathcal{A}$'s advantage between any two consecutive games. And the details of the proof can be found in Appendix C.1.

Theorem 2. *Let $\mathbb{G}$ be a represent group and let $\mathcal{D}$ be a uniformly distributed dictionary of size $|\mathcal{D}|$. Let $\mathcal{P}$ be the improved proposed authentication scheme stated in Section 3. Let $\mathcal{A}$ be an adversary against the semantic security within a time bound t, with less than q_{send} Send-queries and q_{exe} Execution-queries, and making less than q_h random oracle queries. Then we have*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{auth}}_{\mathcal{P}}(\mathcal{A}) &\leq \tfrac{q_{send}}{|\mathcal{D}|} + 5q_h\mathrm{Adv}^{\mathrm{CDH}}_{\mathcal{P}}(t + (q_{send} + q_{exe} + 1)\cdot\tau_e) \\ &\quad + \tfrac{q_h^2+6q_{send}}{2^{l+1}} + \tfrac{(q_{send}+q_{exe})^2}{p},\end{aligned}$$

where $t' \leq t + (q_s + q_p + 1)\tau_e$, τ_e is the computational time for an exponentiation in $\mathbb{G}$ and $l = min\{l_i\}, i = 0, 1, 2, 3$.

Proof. The proof is similar to the previous one. But one can find more details in Appendix C.2.

5.3 Withstand possible attacks

As stated by Menezes [32], a formal security proof isn't an absolute guarantee of security and the old-fashioned cryptanalysis still plays an important (even indispensible) role in establishing and maintaining confidence in the security of a cryptographic protocol. Accordingly, in the following, we discuss some possible attacks presented in related schemes and perform the heuristic security analysis for these attacks on the proposed scheme to verify whether the security requirements mentioned in Section 2 have been satisfied under our adversary model.

The security of our proposed authentication scheme is based on the secure hash function and the discrete logarithm problem. To facilitate the analysis, we define two adversaries:

- $\mathcal{A}_1$: This adversary is with the capabilities of C-0, C-1, C-2(i), C-3 and C-4.
- $\mathcal{A}_2$: This adversary is with the capabilities of C-0, C-1, C-2(ii), C-3 and C-4.

One can easily see that all the adversaries under our adversary model are either $\mathcal{A}_1$ or $\mathcal{A}_2$. Since some high entropy parameters, such as N_i and b, are involved in the authentication process, without them an adversary poses no threat to our scheme. Hence, in the following we only focus on the adversary with capability C-2(ii), that is the adversary $\mathcal{A}_2$.

In particular, with the ability of C-2(ii), the secret information stored in the smart card can be revealed by the adversary $\mathcal{A}_2$, i.e., the security parameters N_i, A_i, b and y can be obtained by a malicious privileged attacker. It is obvious to see that: 1) our scheme can resist against replay attack as fresh nonces are included in the message CID_i, C_3 and C_4; 2) our scheme can resist reflection attack and parallel session attack as the message structures of CID_i, C_3 and C_4 are inherently different; 3)our scheme is free from known key attack as neither the session key $SK = \mathcal{H}_3(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_U)$ is of any part of the transmitted messages nor $\mathcal{A}_2$ can learn any other useful information form SK under the one-way nature of hash functions; 4) our scheme is free from stolen-verifier attack as the user-specific information (ID_i, T_{reg}) stored on the server are not password-involved verifiers; 5)our scheme can achieve forward secrecy as the Diffie-Hellman key exchange technique is employed.

(1) User anonymity: Suppose that the attacker has intercepted U_i's authentication messages $\{CID_i, C_1, C_2, C_3, C_4\}$. Then, the adversary may try to retrieve any static parameter from these messages, but these messages are all session-variant and indeed random strings due to the randomness of u and/or v. Accordingly, without knowing the random number u, the adversary will face to solve the discrete logarithm problem to retrieve the correct value of ID_i from CID_i, while ID_i is the only static element corresponding to U_i in the transmitted messages. Hence, the proposed scheme can preserve identity-protection and unlinkability.

(2) Password disclosure to server: With $\mathcal{H}_0(b \parallel PW_i)$ instead of plaintext password PW_i submitted to server S, it is computationally infeasible to derive PW_i from $h(b \parallel PW_i)$ without knowing the random number b due to the one-way property of the secure hash function.

(3) User impersonation attack: As CID_i, C_3 and C_4 are all protected by secure one-way hash function, any modification to these parameters of the legitimate user U_i's authentication messages will be detected by the server S if the attacker cannot fabricate the valid CID_i, C_3 and C_4. Because the attacker has no way to obtain the value of $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ corresponding to user U_i, he/she cannot fabricate the valid CID_i, C_3 and C_4. Therefore, the proposed protocol is secure against user impersonation attack.

(4) Server masquerading attack: In the proposed protocol, a malicious server MS cannot compute the correct $Y_2 = (C_1)^x \bmod n$ because he/she does not know the value of S's private key x, and thus MS cannot derive the valid $ID_i = CID_i \oplus \mathcal{H}_0(C_1 \parallel Y_2)$. Without knowing U_i's valid ID_i and S's private key x, MS has to break the secure one-way hash function to retrieve $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$. Furthermore, because MS cannot obtain $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$, it is impossible to fabricate the proper $C_3 = \mathcal{H}_1(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_U)$ to pass the verification of U_i in Step $V3$ of the verification phase. Therefore, the proposed protocol is secure against server masquerading attack.

(5) Smart card loss attack(offline password guessing attack): Even after gathering the secret information b, N_k, A_k, n and y , the attacker has to guess both ID_i and PW_i at the same time to launch an offline password guessing attack, because it has been demonstrated that our scheme can provide identity protection. As there is no transcripts over the public channel involving user password, the only way for $\mathcal{A}_2$ to learn some information about PW_i is to find out the correct pair (ID_i^*, PW_i^*) such that $\mathcal{H}_0((\mathcal{H}_0(ID_i^*) \oplus \mathcal{H}_0(PW_i^*)) \bmod n) = A_i$. In Section 3, we have shown that this strategy of $\mathcal{A}_2$ is infeasible as A_i is a "fuzzy verifier". Without correct PW_i, the correct value of $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ cannot be obtained, thus user impersonation attack will not succeed. Thus the proposed scheme can resist smart card loss attack.

(6) Mutual authentication: In our dynamic ID-based scheme, the server authenticates the user by checking the validity of C_4 in the access request. We have shown that our scheme can preserve user anonymity, so user ID_i is only known to the server S and the user U_i itself. We have proved that our scheme can resist user impersonation attack. Therefore, it is impossible for an adversary to forge messages to masquerade as U_i in our scheme. To pass the authentication of server S, the smart card first needs U_i's identity ID_i and password PW_i to get through the verification in Step L2 of the login phase. In this Section, we have shown that our scheme can resist offline password guessing attack. Therefore, only the legal user U_i who owns correct ID_i and PW_i can compute the correct value of $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ to pass the authentication of server S. On the other hand, the user U_i authenticates server S by explicitly checking whether the other party communicating with can compute the valid C_3 or not. Since the malicious server does not know the values of ID_i corresponding to user U_i and x corresponding to server S, only the legitimate server can compute the correct $C_3 = \mathcal{H}_1(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_U)$. From the above analysis, we conclude that our scheme can achieve mutual authentication.

(7) Denial of service attack: Assume that an adversary has got a legitimate user U_i's smart card. However, in our scheme, the user-specific security parameters stored in the server S do not need to be updated during the authentication phase, the risk of in-consistence is eliminated. Furthermore, the smart card checks the validity of user input identity ID_i^* and password PW_i^* before the password update process. Since the smart card computes

$A_i^* = \mathcal{H}_0((\mathcal{H}_0 ID_i^* \oplus \mathcal{H}_0(b \parallel PW_i^*) \bmod n)$ and compares it with the stored value of A_i in its memory to verify the legality of the user before the smart card accepts the password update request. Accordingly, once the number of login failure exceeds the predefined system value, the smart card will be locked immediately. Therefore, denial of service attack is thwarted.

(8) Key compromise impersonation attack: Assume that an adversary $\mathcal{A}$ has got server S's private key x. However, $\mathcal{A}$ cannot compute the correct value of $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$ as she has no knowledge of a legitimate user U_i's registration time T_{reg}. Without the correct value of $\mathcal{H}_0(x \parallel ID_i \parallel T_{reg})$, $\mathcal{A}$ cannot impersonate U_i.

(9) Undetectable online password guessing attack: Assume an adversary $\mathcal{A}$ has got a legitimate user U_i's smart card and managed to obtain the security parameters N_i, A_i and y. Based on this strong assumption, this type of attack is still infeasible. Malicious actions will be detected by S if $\mathcal{A}$ guesses the password online, as S can compute $\mathcal{A}$'s ID in Step V1 of the verification phase and lock the corresponding account after a predefined number of failures. Hence, this type of attack can be effectively thwarted.

6 Performance analysis

To evaluate our scheme, we compare the performance and the satisfaction of the criteria among relevant authentication schemes and our proposed scheme in this section. The reason why the schemes presented in [11, 12, 17, 43, 50, 51], instead of other works mentioned earlier in this paper, are selected to compare with is that, these four schemes are the few ones that can achieve forward secrecy. The criteria of a secure and practical remote user authentication scheme are introduced in Section II, and the comparison results are depicted in Table 3 and 4, respectively.

Table 3. Performance comparison among relevant authentication schemes

	Computation cost	Communication cost	Storage overhead
Our scheme	$6T_E + 12T_H$	2332 bits	3488 bits
Xie [50](2012)	$8T_E + 7T_H$	3328 bits	1280 bits
Wang et al. [43](2012)	$6T_E + 12T_H$	2560 bits	3456 bits
Horng et al. [17](2010)	$7T_E + 4T_S + 8T_H$	2432 bits	3328 bits
Chen et al. [11] (2010)	$6T_E + 5T_H$	2560 bits	3200 bits
Chung et al. [12](2009)	$4T_E + 2T_I + 12T_H$	2560 bits	3200 bits
Xu et al. [51] (2009)	$6T_E + 8T_H$	2816 bits	3200 bits

Since the login phase and verification phase are executed much more frequently than the other two phases, only the computation cost, communication overhead and storage cost during the login phase and verification phase are taken into consideration. Without loss of generality, the security parameter n is assumed to be 32-bit long, the identity ID_i, password PW_i, random numbers, timestamp values and output of secure one-way hash function are all recommended to be

Table 4. Criteria comparison among relevant authentication schemes

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
Our scheme	√	√	√	√	√	√	√	√	√	√	√	√
Xie [50]	√	√	√	×	×	×	√	√	×	√	√	√
Wang [43]	√	√	√	×*	√	√	√	√	√	√	√	√
Horng et al. [17]	√	×	√	√	×**	×	√	√	×	√	√	√
Chen et al. [11]	√	×	×	×	××***	×	√	×	×	√	×	√
Chung et al. [12]	√	×	√	√	√	√	√	√	×	√	×	√
Xu et al. [51]	√	×	×	×	×	×	√	×	×	√	×	√

* As stated in Section 3, the scheme in [43] is vulnerable to an offline password guessing attack once the smart card security is breached under the adversary model illustrated in Table 1.

** In an unpublished work we found Horng et al.'s scheme is prone to undetectable online password guessing attack and key compromise impersonation attack.

*** A reflection attack is identified in their scheme as any adversary can impersonate server S to send $\{M = T, U = V\}$ to U_i on receiving the login request message $\{ID, V, T, N\}$ at any given session.

128-bit long, while n, y and g are all 1024-bit long. Let T_H, T_E, T_I, T_S and T_X denote the time complexity for hash function, exponential operation, inverse operation, symmetric cryptographic operation and XOR operation respectively. Since the time complexity of XOR operation is negligible as compared to the other four operations, we do not take T_X into account. Typically, time complexity associated with these operations can be roughly expressed as $T_E \approx T_I > T_S \geq T_H \gg T_X$ [13].

In our scheme, the parameters $\{b, N_i, A_i, y, g, p, n\}$are stored in the smart card, thus the storage cost is $3488(= 32 + 3 * 128 + 3 * 1024)$bits. The communication overhead includes the capacity of transmitting message involved in the authentication scheme, which is $2332(= 3 * 128 + 2 * 1024)$ bits. During the login and verification phase, the total computation cost of the user and server is $6T_E + 12T_H$. As illustrated in Table 3, the proposed scheme is more efficient than Horng et al.'s scheme and Xie's scheme, enjoys nearly the same performance with Wang et al.'s scheme, Chen et al.'s scheme and Chung et al.'s scheme.

Table 4 gives a comparison of the admired features of our proposed scheme with the other relevant authentication schemes. Our proposed scheme provides the feature of local password update(C2), while the schemes presented by Horng et al., chen et al., Chung et al. and Xu et al. fail to achieve this feature; Our proposed scheme is free from smart card loss problem (C4), while the schemes presented by Xie, Wang et al., Chen et al. and Xu et al. fail to provide this property; Our proposed scheme preserves user anonymity (C11), while the schemes presented by Chen et al., Chung et al. and Xu et al. do not provide this property; our proposed scheme, Wang et al.'s scheme and Chung et al.'s scheme can resist various kinds of known attacks (C5), while the other three latest schemes suffer from several security vulnerabilities; our proposed scheme and Wang et

al.'s scheme provides the property of timely wrong password detection, while the other five schemes fail to achieve it. It is clear that our scheme meets more criteria as compared to other relevant authentication schemes using non-tamper resistant smart cards.

7 Conclusion

In this paper, we have demonstrated damaging attacks on two smart card based password authentication schemes to uncover the subtleties and challenges in designing this type of schemes. As our main contribution, a robust scheme is proposed to remedy these identified flaws, the security and performance analysis demonstrate that our presented scheme achieves all of the twelve independent requirements with high efficiency and thus our scheme is more secure and efficient for practical use. In addition, a practical adversary model and a comprehensive criterion set is proposed to facilitate the design and analysis of this type of schemes in a systematic approach.

Acknowledgments. We sincerely thank Prof. Wang Yongge for quickly and frankly acknowledging the ambiguity in the specification of their scheme PSCAV, and for his invaluable comments and generous advice on our attack and our adversary model. He earns our deep respect. This research was partially supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61170241 and No. 61073042.

References

1. Bao, F., Deng, R.: Privacy protection for transactions of digital goods. In: Qing, S., Okamoto, T., Zhou, J. (eds.) ICICS'01, LNCS, vol. 2229, pp. 202–213. Springer Berlin / Heidelberg (2001)
2. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) Advances in Cryptology-EUROCRYPT 2000, LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
3. Blake-Wilson, S., Johnson, D., Menezes, A.: Key agreement protocols and their security analysis. In: Darnell, M. (ed.) Crytography and Coding, LNCS, vol. 1355, pp. 30–45. Springer Berlin / Heidelberg (1997)
4. Bonneau, J.: The science of guessing: Analyzing an anonymized corpus of 70 million passwords. In: 33rd IEEE Symposium on Security and Privacy (S&P 2012). pp. 538–552. IEEE Computer Society (2012)
5. Boyd, C., Mathuria, A.: Protocols for authentication and key establishment. Springer-Verlag (2003)
6. Bresson, E., Chevassut, O., Pointcheval, D.: Security proofs for an efficient password-based key exchange. In: Proceedings of the 10th ACM conference on Computer and communications security (CCS03). pp. 241–250. ACM, New York, NY, USA (2003)
7. Chang, C.C., Wu, T.C.: Remote password authentication with smart cards. IEE Proceedings-Computers and Digital Techniques 138(3), 165–168 (1991)

8. Chang, C.C., Lee, C.Y.: A secure single sign-on mechanism for distributed computer networks. Industrial Electronics, IEEE Transactions on 59(1), 629–637 (jan 2012)
9. Chen, C.L., Lu, M.S., Guo, Z.M.: A non-repudiated and traceable authorization system based on electronic health insurance cards. Journal of Medical Systems pp. 1–12 (2011), doi:10.1007/s10916-011-9703-4
10. Chen, T., Hsiang, H., Shih, W.: Security enhancement on an improvement on two remote user authentication schemes using smart cards. Future Generation Computer Systems 27(4), 377–380 (2011)
11. Chen, Y.L., Chou, J.S., Huang, C.H.: Improvements on two password-based authentication protocols. Cryptology ePrint Archive, Report 2009/561 (2009), http://eprint.iacr.org/2009/561.pdf
12. Chung, H., Ku, W., Tsaur, M.: Weaknesses and improvement of wang et al.'s remote user password authentication scheme for resource-limited environments. Computer Standards & Interfaces 31(4), 863–868 (2009)
13. Ferguson, N., Schneier, B., Kohno, T.: Cryptography Engineering: Design Principles and Practical Applications. John Wiley & Sons (2010)
14. GROUP, I.W., et al.: Ieee 1363: Standard specifications for public key cryptography. IEEE standard, IEEE, New York, NY 10017 (2005)
15. Halevi, S., Krawczyk, H.: Public-key cryptography and password protocols. ACM Transactions on Information and System Security (TISSEC) 2(3), 230–268 (1999)
16. Hao, F.: On robust key agreement based on public key authentication. In: Sion, R. (ed.) FC'10, LNCS, vol. 6052, pp. 383–390. Springer Berlin / Heidelberg (2010)
17. Horng, W., Lee, C., Peng, J.: A secure remote authentication scheme preserving user anonymity with non-tamper resistant smart cards. WSEAS Transactions on Information Science and Applications 7(5), 619–628 (2010)
18. Huang, X., Xiang, Y., Chonka, A., Zhou, J., Deng, R.: A generic framework for three-factor authentication: preserving security and privacy in distributed systems. Parallel and Distributed Systems, IEEE Transactions on 22(8), 1390–1397 (2011)
19. Kasper, T., Oswald, D., Paar, C.: Side-channel analysis of cryptographic rfids with analog demodulation. In: Juels, A., Paar, C. (eds.) RFIDSec'12, LNCS, vol. 7055, pp. 61–77. Springer Berlin / Heidelberg (2012)
20. Khan, M., Kim, S., Alghathbar, K.: Cryptanalysis and security enhancement of a more efficient & secure dynamic id-based remote user authentication scheme'. Computer Communications 34(3), 305–309 (2011)
21. Klein, D.V.: Foiling the cracker: A survey of, and improvements to, password security. In: Proceedings of the 2nd USENIX Security Workshop. pp. 5–14 (1990)
22. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Advances in Cryptology–CRYPTO99. pp. 789–789. Springer (1999)
23. Krawczyk, H.: Hmqv: A high-performance secure diffie-hellman protocol. In: Shoup, V. (ed.) Advances in Cryptology–CRYPTO 2005, LNCS, vol. 3621, pp. 546–566. Springer Berlin / Heidelberg (2005)
24. Li, C.T., Lee, C.C.: A robust remote user authentication scheme using smart card. Information Technology And Control 40(3), 236–245 (2011)
25. Li, C.T., Lee, C.C., Liu, C.J., Lee, C.W.: A robust remote user authentication scheme against smart card security breach. In: Li, Y. (ed.) DBSec'11, LNCS, vol. 6818, pp. 231–238. Springer Berlin / Heidelberg (2011)
26. Liao, I., Lee, C., Hwang, M.: A password authentication scheme over insecure networks. Journal of Computer and System Sciences 72(4), 727–740 (2006)

27. Ma, C.G., Wang, D., Zhang, Q.M.: Cryptanalysis and improvement of sood et al.s dynamic id-based authentication scheme. In: Ramanujam, R., Ramaswamy, S. (eds.) ICDCIT'12, LNCS, vol. 7154, pp. 141–152. Springer-Verlag (2012)
28. Ma, C.G., Wang, D., Zhao, P., Wang, Y.H.: A new dynamic id-based remote user authentication scheme with forward secrecy. In: Wang, H., Zou, L., Huang, G., He, J., Pang, C., Zhang, H., Zhao, D., Yi, Z. (eds.) APWeb'12, LNCS, vol. 7234, pp. 199–211. Springer Berlin / Heidelberg (2012)
29. Madhusudhan, R., Mittal, R.: Dynamic id-based remote user password authentication schemes using smart cards: A review. Journal of Network and Computer Applications (2012), doi:10.1016/j.jnca.2012.01.007
30. Mangard, S., Oswald, E., Standaert, F.X.: One for all-all for one: unifying standard differential power analysis attacks. IET Information Security 5(2), 100 –110 (june 2011)
31. Mangipudi, K., Katti, R.: A secure identification and key agreement protocol with user anonymity (sika). computers & security 25(6), 420–425 (2006)
32. Menezes, A.: Another look at provable security. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology–EUROCRYPT 2012, Lecture Notes in Computer Science, vol. 7237, pp. 8–8. Springer Berlin / Heidelberg (2012), availe at http://www.cs.bris.ac.uk/eurocrypt2012/Program/Weds/Menezes.pdf
33. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining smart-card security under the threat of power analysis attacks. IEEE Transactions on Computers 51(5), 541–552 (2002)
34. Pu, Q.: An improved two-factor authentication protocol. In: 2010 International Conference on Multimedia and Information Technology (MMIT). vol. 2, pp. 223–226. Ieee (2010)
35. Pu, Q., Wang, J., Zhao, R.: Strong authentication scheme for telecare medicine information systems. Journal of Medical Systems pp. 1–11 (2011), doi:10.1007/s10916-011-9735-9
36. Roy, S., Das, A.K., Li, Y.: Cryptanalysis and security enhancement of an advanced authentication scheme using smart cards, and a key agreement scheme for two-party communication. In: 2011 IEEE 30th International Performance Computing and Communications Conference (IPCCC). pp. 1–7. IEEE (2011)
37. Shim, K.: Security flaws in three password-based remote user authentication schemes with smart cards. Cryptologia 36(1), 62–69 (2012)
38. Song, R.: Advanced smart card based password authentication protocol. Computer Standards & Interfaces 32(5), 321–325 (2010)
39. Sood, S.K.: Secure dynamic identity-based authentication scheme using smart cards. Information Security Journal: A Global Perspective 20(2), 67–77 (2011)
40. Tang, C., Wu, D.: Mobile privacy in wireless networks-revisited. IEEE Transactions on Wireless Communications 7(3), 1035 –1042 (march 2008)
41. Tsai, C., Lee, C., Hwang, M.: Password authentication schemes: current status and key issues. International Journal of Network Security 3(2), 101–115 (2006)
42. Tsai, J., Wu, T., Tsai, K.: New dynamic id authentication scheme using smart cards. International Journal of Communication Systems 23(12), 1449–1462 (2010)
43. Wang, D., Ma, C.G., P., W.: Secure password-based remote user authentication scheme with non-tamper resistant smart cards. In: Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J. (eds.) DBSec'12, LNCS, vol. 7371, pp. 114–121. Springer Berlin / Heidelberg (2012)
44. Wang, R.C., Juang, W.S., Lei, C.L.: Robust authentication and key agreement scheme preserving the privacy of secret key. Computer Communications 34(3), 274–280 (2011)

45. Wang, R.C., Juang, W.S., Lei, C.L.: A simple and efficient key exchange scheme against the smart card loss problem. In: Denko, M., Shih, C.s., Li, K.C., Tsao, S.L., Zeng, Q.A., Park, S., Ko, Y.B., Hung, S.H., Park, J. (eds.) IFIP EUC'07, LNCS, vol. 4809, pp. 728–744. Springer Berlin / Heidelberg (2007)
46. Wang, Y., Liu, J., Xiao, F., Dan, J.: A more efficient and secure dynamic id-based remote user authentication scheme. Computer communications 32(4), 583–585 (2009)
47. Wang, Y.G.: Password protected smart card and memory stick authentication against off-line dictionary attacks. In: Gritzalis, D., Furnell, S., M., T. (eds.) SEC 2012, IFIP AICT, vol. 376, pp. 489–500. Springer Boston (2012), availe at http://coitweb.uncc.edu/ yonwang/papers/smartcard.pdf
48. Wu, S.H., Zhu, Y.F., Pu, Q.: Robust smart-cards-based user authentication scheme with user anonymity. Security and Communication Networks 5(2), 236–248 (2012)
49. Wu, T.: A real-world analysis of kerberos password security. In: Proceedings of the 1999 ISOC Network and Distributed System Security Symposium. pp. 1–14. Internet Soc. (1999)
50. Xie, Q.: Dynamic id-based password authentication protocol with strong security against smart card lost attacks. In: Snac, P., Ott, M., Seneviratne, A., Akan, O. (eds.) Wireless Communications and Applications, LNICST, vol. 72, pp. 412–418. Springer Berlin / Heidelberg (2012)
51. Xu, J., Zhu, W., Feng, D.: An improved smart card based password authentication scheme with provable security. Computer Standards & Interfaces 31(4), 723–728 (2009)
52. Yang, G.M., Wong, D.S., Wang, H.X., Deng, X.T.: Formal analysis and systematic construction of two-factor authentication scheme. In: Ning, P., Qing, S., Li, N. (eds.) ICICS'06, LNCS, vol. 4307, pp. 82–91. Springer Berlin / Heidelberg (2006)
53. Yang, G.M., Wong, D.S., Wang, H.X., Deng, X.T.: Two-factor mutual authentication based on smart cards and passwords. Journal of Computer and System Sciences 74(7), 1160–1172 (2008)
54. Yeh, K.H., Su, C., Lo, N.W., Li, Y., Hung, Y.X.: Two robust remote user authentication protocols using smart cards. Journal of Systems and Software 83(12), 2556–2565 (2010)
55. Yoon, E., Ryu, E., Yoo, K.: Further improvement of an efficient password based remote user authentication scheme using smart cards. Consumer Electronics, IEEE Transactions on 50(2), 612–614 (2004)

A Cryptanalysis of Yang et al.'s Scheme

In ICICS'06, Yang et al. [52] proposed a generic construction framework to convert the conventional provably secure PAKE protocols to smart-card-based versions and further proposed a new scheme to demonstrate its effectiveness. Their new scheme is claimed to be secure and can satisfy all their proposed criteria. In the following, we will show that their scheme is actually vulnerable to an offline password guessing attack, thereby contradicting the claim made in [52] that the new scheme is secure even if the secret data stored in smart card is revealed by the adversary.

Here, we just follow the original notations in [52] as closely as possible. Yang et al.'s scheme consists of four phases: the registration phase, the login-and-authentication phase and the password-changing activity.

Notations. Let G be a subgroup of prime order q of a multiplicative group $\mathcal{Z}_p^*$. Let g be a generator of G. Let (PK_S, SK_S) denote a public/private key pair of the server S. Besides a public/private key pair (PK_S, SK_S), the server S also maintains a long-term secret x which is a random string of length k. Let $H : \{0,1\}^* \rightarrow \{0,1\}^k$ denote a collision resistant hash function and $PRF_K : \{0,1\}^k \rightarrow \{0,1\}^k$ a pseudorandom function keyed by K.

Registration phase. Server S issues a client A as follows.:

1. A arbitrarily chooses a unique identity ID_A and sends it to S.
2. S calculates $B = PRF_x(H(ID_A)) \oplus H(PW_0)$ where PW_0 is the initial password (e.g. a default such as a string of all '0').
3. S issues A a smart card which contains PK_S, ID_A, B, p, g, q. In practice, we can have all these parameters except B be burned in the read-only memory of the smart card when the smart card is manufactured.
4. On receiving the smart card, A changes the password immediately by performing the password-changing activity (described below).

Login-and-authentication phase. When A wants to login to S, she attaches the smart card to a card reader, and then keys ID_A and PW_A. The smart card checks if the input identity is equal to the value stored in it. If not, the login request is rejected. Otherwise, the smart card retrieves the value $LPW = B \oplus H(PW_A)$. A (actually performed by the client's smart card) and S then use LPW as the password to perform the PWAKE protocol:

1. A first chooses a random number $x \in \mathcal{Z}_p^*$, computes g^x and then sends (A, sid, g^x) to S.
2. Upon receiving the login request, S also chooses a random number $y \in \mathcal{Z}_p^*$, computes $\tau = SIG_{SK_S}(S, A, sid, g^x, g^y)$ and sends (S, sid, g^y, τ) to A.
3. S Upon receiving (S, sid, g^y, τ), A first checks the validity of τ using PK_S. If it is valid, A computes $c = ENC_{PK_S}(LPW_A, A, S, sid, g^x, g^y)$ and sends (A, sid, c) to S.
4. Upon receiving (A, sid, c), S first decrypts c using its private key SK_S and gets $(LPW_A, A, S, sid, g^x, g^y)$. S then checks whether: 1) the decrypted g^x, g^y are the same as previously sent by A and S respectively; 2) the decrypted LPW_A is equal to $PRF_x(H(ID_A))$. If both conditions are satisfied, then S confirms that A is a valid user.
5. In the end, A and S agree on the common session key calculated as $\sigma = g^{xy}$.

Password-changing activity. If A wants to change the password, the following steps is carried out:

1. A select a new password PW'_A.
2. Compute $B' = B \oplus H(PW_A) \oplus H(PW'_A)$, where PW_A is the old password.
3. Replace B with B' in the smart card.

A.1 Offline password guessing attack

In [52] and its full version [53], it is claimed that the scheme is secure against offline password guessing attack even if A's smart card is compromised (i.e., the

secret data stored in the card is revealed). The scheme is based on a proven secure PAKE protocol proposed by Halevi and Krawczyk [15] and itself is also armed with a formal proof. Surprisingly, the claimed goal cannot be achieved, while the security flaw is rather obvious: adversaries can easily get useful information about the password from the transcripts of protocol runs. Suppose an adversary $\mathcal{A}$ has compromised A's smart card and obtained the stored secret B, with the previously intercepted $\{A, S, g^x, g^y, sid, c\}$ from the public channel, the attack can be carried out by $\mathcal{A}$ to obtain A's password PW_A as follows:

Step 1. Guesses the value of PW_A to be PW_A^* from dictionary space $\mathcal{D}$.
Step 2. Computes $LPW_A^* = H(PW_A^*) \oplus B$.
Step 3. Computes $c^* = ENC_{PK_S}(LPW_A^*, A, S, sid, g^x, g^y)$, where $\{A, S, g^x, g^y, sid\}$ is previously intercepted from the public channel during one of A's login-and-authentication phase and PK_S is a public parameter.
Step 4. Verifies the correctness of PW_A^* by checking if the computed c^* is equal to the intercepted c.
Step 5. Repeats Step 1, 2, 3, and 4 of this procedure until the correct value of PW_A is found.

As the size of the password dictionary, i.e. $|\mathcal{D}|$, is very limited in practice, the above attack procedure can be completed in polynomial time.

Notes. We have analyzed more than eighty recent smart-card-based password authentication schemes and find one third of them vulnerable to a similar attack as described above, which highlights the difficulties in designing such schemes. Particularly, all of the schemes based on conventional PAKE protocols are either vulnerable to password guessing attack or insecure in password-change-activity, i.e. none can fulfill the criterion C4, which highlights the differences from conventional PAKE protocols. Besides the scheme just described above, another typical example is Wang's two schemes, namely PSCAV and PSCA. Both PSCAV and PSCA is prone to a similar attack as described above, while PSCAV are based on an identity-based key agreement protocol from IEEE 1363.3 [14] and PSCA is based on the well-known HMQV [23].

B Cryptanalysis of PSCAV

In SEC'12, Wang [47] observed that the previous papers in this area present attacks on protocols in previous papers and propose new protocols without proper security justification (or even a security model to fully identify the practical threats), which contributes to the main cause of the above failure. Accordingly, Wang presented three kinds of security models, namely Type I, II and III, and further proposed four concrete schemes, only two of which, i.e. PSCAb and PSCAV , are claimed to be secure under the harshest model, i.e. Type III security model. However, PSCAb requires Weil or Tate pairing operations to defend against offline guessing attack and may not be suitable for systems where pairing operations are considered to be too expensive or infeasible to implement. Moreover, PSCAb suffers from the well-known key escrow problem and lacks some

desirable features such as local password update, repairability and user anonymity. As for PSCAV, in this paper, we will demonstrate that it still cannot achieve the claimed security goals and is vulnerable to an offline password guessing attack and other attacks under the Type III security model in the following.

B.1 A brief review of PSCAV

In this section, we firstly give a brief review of PSCAV and then present the attack. Here we just follow the original notations in [47] as closely as possible. Assume that the server has a master secret β (β could be user specific also). For each user with password α, let the user specific generator be $g_{\mathcal{C}} = \mathcal{H}(\mathcal{C}, \alpha, \beta)$, the value $g_{\mathcal{C}}^{\mathcal{H}_2(\alpha)}$ is stored in the smart card, where $\mathcal{H}_2$ is another independent hash function. The value $g_{\mathcal{C}} = \mathcal{H}(\mathcal{C}, \alpha, \beta)$ will be stored in the server's database for this user. The remaining of the protocol runs as follows:

1. The card selects random x and sends $R_A = g_{\mathcal{C}}^x$ to the server;
2. Server selects random y and sends $R_A = g_{\mathcal{C}}^y$ to the card;
3. The card computes $u = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_A, R_B)$ and $sk = g_{\mathcal{C}}^{y(x+u)}$, where S is identity string of the server and $g_{\mathcal{C}} = \mathcal{D}_{\mathcal{H}_2(\alpha)}(g_{\mathcal{C}}^{\mathcal{H}_2(\alpha)})$;
4. The card sends $C_{\mathcal{C}} = H(sk, \mathcal{C}, \mathcal{S}, R_A, R_B, 1)$ to the server;
5. After verifying that $C_{\mathcal{C}}$ is correct, server computes $u = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_A, R_B)$, $sk = g_{\mathcal{C}}^{y(x+u\alpha)} = (g_{\mathcal{C}}^x)^y \cdot g_{\mathcal{C}}^{yu\alpha} = (R_A)^y \cdot g_{\mathcal{C}}^{yu\alpha}$, and sends $C_{\mathcal{S}} = \mathcal{H}(sk, \mathcal{S}, \mathcal{C}, R_B, R_A, 2)$ to the card.

Since the session key sk is computed with the contribution of the password α by server S in the above Step 5, the password α (or the parameter g_C^α) is needed to be known by S. However, the original specification in [47] does not explicitly explain how can the server obtain the user's password α to compute sk in the above Step 5. We assume g_C^α is also stored in the server's database, i.e. an entry $(\mathcal{C}, g_{\mathcal{C}}, g_C^\alpha)$ corresponding to user $\mathcal{C}$ is stored in the server's database.[5] This ambiguity does not affect our security analysis however.

B.2 Offline password guessing attack

In [47], it is claimed that PSCAV is secure in Type III security model, i.e., the scheme can withstand offline password guessing attack even if C's smart card is compromised (i.e., the secret data stored in the card is revealed). In the following, we demonstrate that this is not true. Suppose an adversary $\mathcal{A}$ has compromised C's smart card and obtained the stored secret $g_{\mathcal{C}}^{\mathcal{H}_2(\alpha)}$. The attack can be carried out by $\mathcal{A}$ to obtain C's password α as follows:

Step 1. On intercepting $R_A = g_{\mathcal{C}}^x$ from client $\mathcal{C}$, $\mathcal{A}$ blocks it and sends $R_B = g_{\mathcal{C}}^{\mathcal{H}_2(\alpha)}$ to the client on behalf of the server;

[5] This ambiguity and the corresponding remedy have been confirmed by the author of [47], and he earns our deep respect for his frankly and quickly acknoledgement.

Step 2. On receiving the response $C_{\mathcal{C}}$, $\mathcal{A}$ computes $u = \mathcal{H}(\mathcal{C}, \mathcal{S}, R_A, R_B)$.
Step 3. Guesses the value of password α to be α^* from dictionary space $\mathcal{D}$.
Step 4. Computes $g_{\mathcal{C}}^* = \mathcal{D}_{\mathcal{H}_2(\alpha^*)}(g_{\mathcal{C}}^{\mathcal{H}_2(\alpha)})$;
Step 5. Computes $sk^* = g_{\mathcal{C}}^{x\mathcal{H}_2(\alpha^*)} \cdot (g_{\mathcal{C}}^*)^{u\alpha^*\mathcal{H}_2(\alpha^*)}$
$= (R_A)^{\mathcal{H}_2(\alpha^*)} \cdot (g_{\mathcal{C}}^*)^{u\cdot\alpha^*\cdot\mathcal{H}_2(\alpha^*)}$;
Step 6. $\mathcal{A}$ computes $C_{\mathcal{C}}^* = H(sk^*, \mathcal{C}, \mathcal{S}, R_A, R_B, 1)$;
Step 7. Verifies the correctness of α^* by checking if the computed $C_{\mathcal{C}}^*$ is equal to the received $C_{\mathcal{C}}$;
Step 8. Repeats the above Steps 3-8 until the correct value of α is found.

As the size of the password dictionary, i.e. $|\mathcal{D}|$, is very limited in practice [21], e.g. $|\mathcal{D}| = 10^6$, $\mathcal{A}$ may recover the password in seconds on a PC by a single run of the PSCAV protocol.

B.3 Other security drawbacks

In many cases, it is of utmost importance to provide user anonymity so that the adversary cannot trace user activity. For example, in e-commerce applications, once the identity of the user is leaked, the sensitive information such as shopping patterns, individual preferences, etc., can be learned and abused for marketing purposes [1]. In addition, the leakage of the user identity may also cause an unauthorized entity to track the user's login history and current location [40]. Therefore, user anonymity is an glamorous and important feature that a practical authentication scheme should achieve [27, 29, 39, 41]. However, in PSCAV, the user's identity ID is transmitted in plain, which may leak the identity of the logging user once the login messages were eavesdropped. In other words, without employing any effort an adversary can recognize the particular transaction being performed by the user C. Moreover, the user's identity ID is static in all the login phases, which may facilitate the attacker to trace out the different login request messages belonging to the same user and to derive some information related to the user C. In a word, neither identity protection nor user un-traceability can be preserved in PSCAV.

Since user C's critical parameter is computed as $g_{\mathcal{C}} = \mathcal{H}(\mathcal{C}, \alpha, \beta)$, it is not easily repairable once $g_{\mathcal{C}}$ is leaked out, which can happen in various ways, such as a break of password just as stated in the previous section.

C Formal security analysis of our scheme

C.1 Proof of Theorem 1

Proof. In the proof below, we do not consider forward-secrecy for simplicity. We incrementally define a sequence of games starting at the real attack game G_0 and ending up with G_8. For each game G_n (n=0,1, ..., 8), we define the following events:

– $\mathbf{Succ_n}$ occurs if $\mathcal{A}$ correctly guesses the bit c involved in the Test-query.

- **AskPara$_\mathbf{n}$** occurs if $\mathcal{A}$ correctly computes the parameter k by asking a hash query $\mathcal{H}_0$ on $b\|PW_i$ or $x\|ID_i\|T_{reg}$.
- **AskAuth$_\mathbf{n}$** occurs if $\mathcal{A}$ correctly computes the parameter k and asks a hash query $\mathcal{H}_1$ (or $\mathcal{H}_2$) on $ID_i\|ID_S\|Y_1\|C_2\|k\|K$, where K is K_U or K_S.
- **AskH$_\mathbf{n}$** occurs if the adversary asks a hash query $\mathcal{H}_i$ $(i = 1, 2, 3)$ on $ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K$, where K is K_U or K_S.

Game G$_\mathbf{0}$: This game corresponds to the real attack, in the random oracle model. By definition, we have

$$\mathrm{Adv}_{\mathcal{P}}^{\mathrm{ake}}(\mathcal{A}) = 2\Pr[\mathrm{Succ}_0] - 1. \tag{1}$$

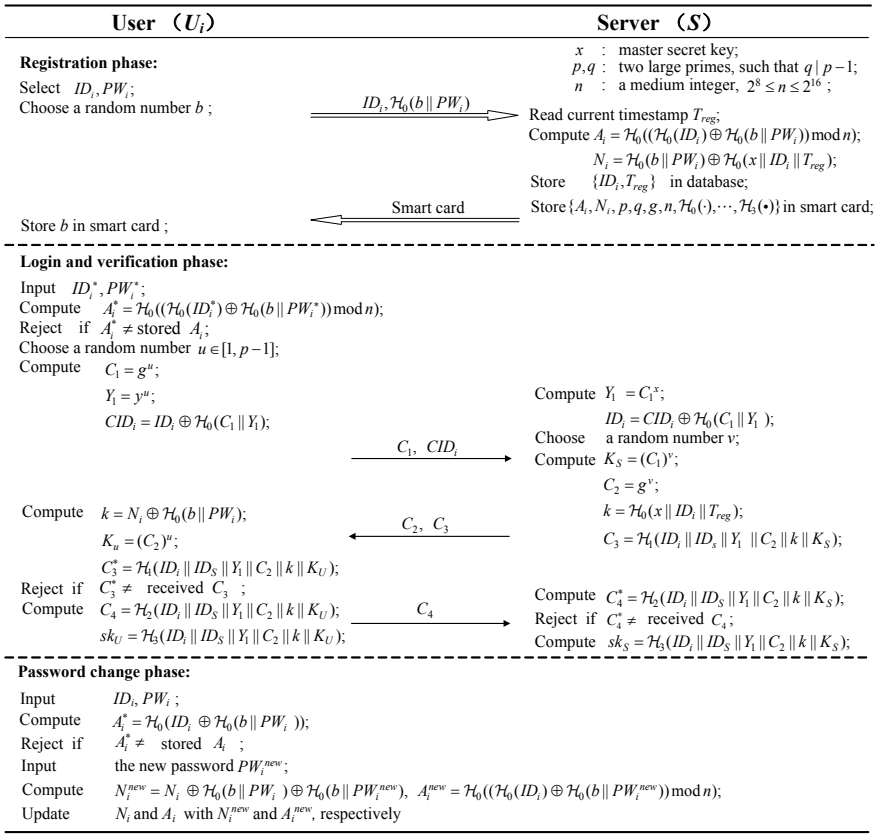

Fig. 1. The proposed scheme

Game G$_\mathbf{1}$: In this game, we simulate the hash oracles $\mathcal{H}_i$(i=0,1,2 and 3, but also four additional hash functions $\mathcal{H}_i'$ that will appear in Game **G$_\mathbf{7}$**) as usual by maintaining a hash list $\Lambda_{\mathcal{H}}$(and another list $\Lambda_{\mathcal{A}}$ containing the hash-queries

asked by the adversary itself). We also simulate all the instances, as the real players would do, for the Send-queries and for the Execute, Reveal, Corrupt and Test-queries (see Figure 2).

For a hash-query $\mathcal{H}_i(q)$ or $\mathcal{H}_i'(q)$ (with $i \in \{0,1,2,3\}$), such that a record (i,q,r) appears in $\Lambda_{\mathcal{H}}$, the answer is r. Otherwise the answer r is defined according to the following rule:

▶ **Rule** $\mathcal{H}^{(1)}$ — Choose a random element $r \in \{0,1\}^{\ell_i}$.
The record(i,q,r) Is added to $\Lambda_{\mathcal{H}}$. If the query is directly asked by the adversary, one adds(i,q,r) to $\Lambda_{\mathcal{A}}$.

We answer to the Send-queries to the client as follows:

— A Send (U^i, Start) -query is processed according to the following rule:
▶ **Rule U1**$^{(1)}$ — Choose $\theta \in_R \mathbb{Z}_p^*$ and compute $C_1 = g^\theta$, $Y_1 = y^\theta$ and $CID_i = ID_i \oplus \mathcal{H}_0(C_1 \| Y_1)$.
Then the query is answered with (C_1, CID_i), and the client instance goes to an expected state.
— If the client instance U^i is in an expected state, a query Send $(U^i,(C_2,C_3))$ is processed by computing the session key and producing an authenticator. We apply the following rules:
▶ **Rule U2**$^{(1)}$ — Compute $k = N_i \oplus \mathcal{H}_0(b \| PW_i)$, $K_U = C_2{}^\theta$ and $C_3^* = \mathcal{H}_1(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_U)$. Reject if the computed C_3^* is not equal to the received C_3. Otherwise moves on.
▶ **Rule U3**$^{(1)}$ — Compute the authenticator $C_4 = \mathcal{H}_2(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_U)$ and the session key $sk_U = \mathcal{H}_3(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_U)$.
Finally the query is answered with C_4, the client instance accepts and terminates. Our simulation also adds $((C_1, CID_i),(C_2,C_3),C_4)$ to Λ_{Ψ}. The variable Λ_{Ψ} keeps track of the exchanged messages.

We answer to the Send-queries to the server as follows:
— A Send $(S^j,(C_1,CID_i))$-query is processed according to the following rule:
▶ **Rule S1**$^{(1)}$ — Compute $Y = C_1{}^x$ and $ID_i = CID_i \oplus \mathcal{H}_0(C_1 \| Y_1)$.
Choose a random exponent $\varphi \in \mathbb{Z}_q^*$ compute $K_S = C_1{}^\varphi$, $C_2 = g^\varphi$ and $k = \mathcal{H}_0(x \| ID_i \| T_{reg})$.
▶ **Rule S2**$^{(1)}$ — Compute $C_3 = \mathcal{H}_1(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_S)$.
Finally, the query is answered with (C_2, C_3) and the server instance goes to an expected state.
— If the server instance S^j is in an expected state, a query Send (S^j, C_4) is processed according to the following rules:
▶ **Rule S3**$^{(1)}$ — Compute $C_4^* = \mathcal{H}_2(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_S)$, and check whether $C_4^* = C_4$.
If the equality does not hold, the server instance terminates without accepting.
If equality holds, the server instance accepts and goes on, applying the following rule:
▶ **Rule S4**$^{(1)}$ — Compute the session key $sk_S = \mathcal{H}_3(ID_i \| ID_S \| Y_1 \| C_2 \| k \| K_S)$.
Finally, the server instance terminates.

An Execute (U^i, S^j)-query is processed using a successive simulations of the Send-queries: $(U,(C_1,CID_i)) \leftarrow$ Send (U^i, Start), $(C_2,C_3) \leftarrow$ Send $(S^j,(C_1,CID_i))$ and $C_4 \leftarrow$ Send $(U^i,(C_2,C_3))$, and outputting the transcript $((C_1,CID_i),(C_2,C_3),C_4)$.

A Corrupt(I, a)-query returns the password PW_i if a=1, returns $\{N_i, A_i, b\}$ if a=2. (Since we do not consider forward secrecy in this proof, no Corrupt(I, 3) occurs.

A Reveal(I)-query returns the session key(sk_U or sk_S) computed by the instance I (if the latter has accepted).

A Test(I)-query first gets sk from Reveal(I), and flips a coin c. If c = 1, we return the value of the session key sk, otherwise we return a random value drawn from $\{0,1\}^{\ell_3}$.

Fig. 2. Simulation of the queries in our scheme

From this simulation, one can easily see that this game is perfectly indistinguishable from the real attack. Hence,

$$\left|\Pr[\text{Succ}_1] - \Pr[\text{Succ}_0]\right| = 0 \tag{2}$$

Game $\mathbf{G_2}$: For an easier analysis, in this game, we simulate all oracles as in game $\mathbf{G_1}$ except that we cancel games in which some (unlikely) collisions appear:

- collisions on the partial transcripts $((C_1, CID_i), (C_2, C_3), C_4)$. Note that transcripts involve at least one honest party, and thus one of C_1 or C_2 is truly uniformly distributed;
- collisions on the output of hash queries.

Both probabilities are bounded by the birthday paradox:

$$\left|\Pr[\text{Succ}_2] - \Pr[\text{Succ}_1]\right| \leq \frac{(q_{send} + q_{exe})^2}{2p} + \frac{q_h^2}{2^{l+1}} \tag{3}$$

where $l = min\{l_i\}, i = 0, 1, 2, 3$.

Game $\mathbf{G_3}$: We define game $\mathbf{G_3}$ by aborting the game wherein the adversary may have lucky in guessing the correct parameter k. We reach this aim by modifying the way the participants processes the queries. We use the rule as follows:

- **Rule U3$^{(3)}$** – Look for a record $(0, *\|ID_i\|*, k)$ in $\Lambda_{\mathcal{A}}$. If such a record does not exist, we abort the game. Otherwise, compute $C_4 = \mathcal{H}_2(ID_i\|ID_S\|Y_1\| C_2\|k\|K_U)$ and the session key $sk_U = \mathcal{H}_3(ID_i\|ID_S\|Y_1\|C_2\|k\|K_U)$.
- **Rule S3$^{(3)}$** – computes $C_4^* = \mathcal{H}_2(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_S)$ and then checks if C_4^* equals the received value of C_4. If this verification holds, the server looks for a record $(0, *, P_i)$ in $\Lambda_{\mathcal{A}}$ and a record $((C_1, CID_i), (C_2, C_3), C_4)$ in Λ_{Ψ}. If such records do not exist, we abort the game.

Since C_1 and C_2 did not appear in a previous session (since the Game $\mathbf{G_2}$), this happens only if the parameter k had been correctly guessed by the adversary without asking $\mathcal{H}_0$:

$$\left|\Pr[\text{Succ}_3] - \Pr[\text{Succ}_2]\right| \leq \frac{q_{send}}{2^{l_0}} \tag{4}$$

Game $\mathbf{G_4}$: We define this game by aborting the game wherein the adversary may have computed the correct parameter k and impersonate as a client or server. We reach this aim by modifying the way the participants processes the queries. We use the rule as follows:

- **Rule U2$^{(4)}$** – Compute $k = \mathcal{H}_0(x \parallel ID_i \parallel T_{reg}) = N_i \oplus \mathcal{H}_0(b \parallel PW_i)$, look for a record $(0, *\|ID_i\|*, k)$ in $\Lambda_{\mathcal{A}}$. If such a record exists, we abort the game. Otherwise, compute $K_U = (C_2)^u \bmod p$, $C_3^* = \mathcal{H}_1(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_U)$, and compare C_3^* with the received C_3. If the equality doesnot hold, terminate without acceptance. Otherwise, move on.
- **Rule S3$^{(4)}$** – computes $C_4^* = \mathcal{H}_2(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_S)$ and then checks if C_4^* equals the received value of C_4. If this verification holds, the server looks for a record $(0, *, P_i)$ in $\Lambda_{\mathcal{A}}$ and a record $((C_1, CID_i), (C_2, C_3), C_4)$ in Λ_{Ψ}. If such records exist, we abort the game.

The two games $\mathbf{G_4}$ and $\mathbf{G_3}$ are perfectly indistinguishable unless the event $\mathbf{AskPara_4}$ occurs:

$$\left|\Pr[\text{Succ}_4] - \Pr[\text{Succ}_3]\right| \leq \Pr[\text{AskPara}_4] \tag{5}$$

Game $\mathbf{G_5}$: We define game $\mathbf{G_5}$ by aborting the game wherein the adversary may have lucky in guessing the correct authenticator C_3 or C_4 (that is, without asking the corresponding hash query $\mathcal{H}_1$ or $\mathcal{H}_2$). Since C_1 and C_2 did not appear in a previous session (since the Game $\mathbf{G_2}$), this happens only if the authenticator C_3 (or C_4) had been correctly guessed by $\mathcal{A}$ without asking $\mathcal{H}_1$ (or $\mathcal{H}_2$):

$$\left|\Pr[\text{Succ}_5] - \Pr[\text{Succ}_4]\right| \leq \frac{q_{send}}{2^l} \quad \left|\Pr[\text{AskPara}_5] - \Pr[\text{AskPara}_4]\right| \leq \frac{q_{send}}{2^l} \tag{6}$$

Game $\mathbf{G_6}$: We define this game by aborting the game wherein the adversary may have computed the correct authenticator C_3 or C_4 (that is, by asking the corresponding hash query $\mathcal{H}_1$ or $\mathcal{H}_2$) and impersonate as a client or server. We reach this aim by modifying the way the participants processes the queries. We use the rule as follows:

- **Rule U3$^{(6)}$** – Check if $(1, ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel *, C_3) \in \Lambda_{\mathcal{A}}$. If it holds, we abort the game. Otherwise, the user goes on to compute C_4 and sk_U.
- **Rule S3$^{(6)}$** – computes $C_4^* = \mathcal{H}_2(ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel K_S)$ and then checks if C_4^* equals the received value of C_4. If this verification holds, the server looks for a record $(0, *, P_i)$ or $(2, ID_i \parallel ID_S \parallel Y_1 \parallel C_2 \parallel k \parallel *, C_4)$ in $\Lambda_{\mathcal{A}}$, and a record $((C_1, CID_i), (C_2, C_3), C_4)$ in Λ_{Ψ}. If such records exist, we abort the game.

The two games $\mathbf{G_6}$ and $\mathbf{G_5}$ are perfectly indistinguishable unless event $\mathbf{AskAuth_6}$ occurs:

$$\left|\Pr[\text{Succ}_6] - \Pr[\text{Succ}_5]\right| \leq \Pr[\text{AskAuth}_6] \tag{7}$$

$$\left|\Pr[\text{AskPara}_6] - \Pr[\text{AskPara}_5]\right| \leq \Pr[\text{AskAuth}_6] \tag{8}$$

Game $\mathbf{G_7}$: In this game, we replace the random oracles $\mathcal{H}_i$ with the private oracles $\mathcal{H}_i'(i = 1, 2, 3)$:

$$\begin{aligned} &C_3 = \mathcal{H}_1'(ID_i \parallel ID_S \parallel C_1 \parallel C_2);\\ &C_4 = \mathcal{H}_2'(ID_i \parallel ID_S \parallel C_1 \parallel C_2);\\ &sk_U = sk_S = \mathcal{H}_3'(ID_i \parallel ID_S \parallel C_1 \parallel C_2) \end{aligned}$$

As a result, the values of C_3, C_4, sk_U, sk_S are completely independent from k, K_U and K_S. $\mathbf{G_7}$ and $\mathbf{G_6}$ are indistinguishable unless the event $\mathbf{AskH_7}$ occurs:

$$\left|\Pr[\text{Succ}_7] - \Pr[\text{Succ}_6]\right| \leq \Pr[\text{AskH}_7] \tag{9}$$

$$\left|\Pr[\text{AskPara}_7] - \Pr[\text{AskPara}_6]\right| \leq \Pr[\text{AskH}_7] \tag{10}$$

$$\left|\Pr[\text{AskAuth}_7] - \Pr[\text{AskAuth}_6]\right| \leq \Pr[\text{AskH}_7] \tag{11}$$

Lemma 1. *The probabilities of the events* $\mathbf{Succ_7}$ *and* $\mathbf{AskPara_7}$ *in this game can be up-bounded by the following values:*

$$\left|\Pr[\text{Succ}_7]\right| = \frac{1}{2} \tag{12}$$

$$\left|\Pr[\text{AskPara}_7]\right| \leq \frac{q_{send}}{|\mathcal{D}|} + \frac{q_{send}}{2^{l_0}} \tag{13}$$

Proof. In the game $\mathbf{G_7}$, the session keys are computed with private hash oracle unknown to $\mathcal{A}$, and thus $\Pr[\text{Succ}_7] = \frac{1}{2}$.

Let us denote by $R(U)$ the set of (C_2, C_3) received by a client instance, and by $R(S)$ the set of C_4 used by a server instance. Since we have avoided the cases where $\mathcal{A}$ have been lucky in guessing k, $\mathcal{A}$ can correctly computed k with the help of either a Corrupt$(I = U^i, 1)$-query or a Corrupt$(I = U^i, 2)$-query, the probability of which is denoted by $\Pr[\text{AskPara}_7\text{WithCorr}_1]$ and $\Pr[\text{AskPara}_7\text{WithCorr}_2]$, respectively. From an information theoretical point of view, since we have avoided collisions in the Game $\mathbf{G_2}$,

$$\begin{aligned}\left|\Pr[\text{AskPara}_7\text{WithCorr}_1]\right| &= \Pr_{pw}[\exists C_3 \in R(U), (1, ID_i \| ID_S \| Y_1 \| C_2 \| k \| *, C_3) \in \Lambda_{\mathcal{A}}] \\ &\quad + \Pr_{pw}[\exists C_4 \in R(S), (0, *, P_i) \in \Lambda_{\mathcal{A}}] \\ &\leq \tfrac{\#R(S)+\#R(U)}{|\mathcal{D}|} = \tfrac{q_{send}}{|\mathcal{D}|}\end{aligned} \tag{14}$$

$$\left|\Pr[\text{AskPara}_7\text{WithCorr}_2]\right| \leq \tfrac{q_{send}}{2^{l_0}} \tag{15}$$

Game $\mathbf{G_8}$: In this game, we simulate the executions using the random self-reducibility of the Diffie-Hellman problem, given one CDH instance (A, B). We do not need to known the values of θ and φ, since the values of K_U and K_S are no longer needed to compute the authenticators and the session keys:

- **Rule U1$^{(8)}$** – chooses a random number $\alpha \in Z_p^*$ and computes $C_1 = A^\alpha \bmod p$, $Y_1 = y^\alpha \bmod p$, $CID_i = ID_i \oplus \mathcal{H}_0(C_1 \parallel Y_1)$. Also add the record (C_1, α) in Λ_A.
- **Rule S1$^{(8)}$** – computes $Y_2 = (C_1)^x \bmod p$ using its private key x, and derives $ID_i = CID_i \oplus \mathcal{H}_0(C_1 \parallel Y_2)$ and checks whether ID_i is in the correct format. If ID_i is not valid, the session is terminated. Otherwise, the server chooses a random number $\beta \in Z_p^*$ and computes $C_2 = B^\beta$. Also add the record (C_2, β) in Λ_B.

$$\Pr[\text{AskH}_8] = \Pr[\text{AskH}_8] \tag{16}$$

Remember that $\mathbf{AskH_8}$ means that the adversary $\mathcal{A}$ had queried the random oracles $\mathcal{H}_i (i = 1, 2, 3)$ on $(ID_i \parallel ID_S \| Y_1 \| C_2 \| k \| CDH(C_1, C_2))$. By picking randomly in the $\Lambda_{\mathcal{A}}$-list we can get the Diffie-Hellman secret value with probability $\frac{1}{q_h}$. This is a triple $(C_1, C_2, \text{CDH}(C_1, C_2))$. We can then simply look in the lists Λ_A and Λ_B to find the values α and β such that $C_1 = A^\alpha$ and $C_2 = B^\beta$:

$$\text{CDH}(C_1, C_2) = \text{CDH}(A^\alpha, B^\beta) = \text{CDH}(A, B)^{\alpha\beta},$$

and thus:

$$\left|\Pr[\text{AskH}_8]\right| \leq q_h \text{Adv}_{\mathcal{P}}^{\text{CDH}}(t') \tag{17}$$

where $t' \leq t + (q_{send} + q_{exe} + 1) \cdot \tau_e$.

Conclusion of the proof: By combining all the above equations, one gets the announced result. Firstly, from Eqs.(1)-(5) we get:

$$\left|\Pr[\text{Succ}_4] - \Pr[\text{Succ}_0]\right| \leq \frac{(q_{send}+q_{exe})^2}{2p} + \frac{q_h^2}{2^{l+1}} + \frac{q_{send}}{2^l} + \Pr[\text{AskPara}_4].$$

Secondly, from Eqs.(6)-(9) we get:

$$\left|\Pr[\text{Succ}_7] - \Pr[\text{Succ}_4]\right| \leq \frac{q_{send}}{2^l} + \Pr[\text{AskAuth}_6] + \Pr[\text{AskH}_7].$$

Finally, from definition we know $\Pr[\text{AskAuth}_6] \leq \Pr[\text{AskH}_6]$:

$$\begin{aligned}\text{Adv}_{\mathcal{P}}^{\text{ake}}(\mathcal{A}) &= 2\Pr[\text{Succ}_7] - 1 + 2(\Pr[\text{Succ}_7] - \Pr[\text{Succ}_0]) \\ &\leq \frac{2q_{send}}{|\mathcal{D}|} + 8q_h \text{Adv}_{\mathcal{P}}^{\text{CDH}}(t + (q_{send} + q_{exe} + 1) \cdot \tau_e) \\ &\quad + \frac{q_h^2+6q_{send}}{2^l} + \frac{(q_{send}+q_{exe})^2}{p},\end{aligned}$$

where $t' \leq t + (q_{send} + q_{exe} + 1) \cdot \tau_e$ and $l = min\{l_i\}, i = 0, 1, 2, 3$.

C.2 Proof of Theorem 2

Proof. The proof is similar to that of Theorem 1.

Firstly, we define an additional event:

– $\mathbf{Auth_n}$ occurs if $\mathcal{A}$ correctly guesses the authenticator C_3 or C_4 that will be accepted by the corresponding party and that has been built by the adversary herself in game $\mathbf{G_n}, n = 0, 1, \ldots, 7$.

Thus, we define

$$\text{Adv}_{\mathcal{P}}^{\text{auth}}(\mathcal{A}) = \Pr[\text{Auth}_0]$$

Secondly, we use the same sequence of games presented in the previous section, and extend Eqs. (1)-(7) to obtain:

$$\begin{aligned}&\left|\Pr[\text{Auth}_1] - \Pr[\text{Auth}_0]\right| = 0 \\ &\left|\Pr[\text{Auth}_2] - \Pr[\text{Auth}_1]\right| \leq \frac{(q_{send} + q_{exe})^2}{2p} + \frac{q_h^2}{2^{l+1}} \\ &\left|\Pr[\text{Auth}_3] - \Pr[\text{Auth}_2]\right| \leq \frac{q_{send}}{2^l} \\ &\left|\Pr[\text{Auth}_4] - \Pr[\text{Auth}_3]\right| \leq \Pr[\text{AskPara}_4] \\ &\left|\Pr[\text{Auth}_5] - \Pr[\text{Auth}_4]\right| \leq \frac{q_{send}}{2^l} \\ &\left|\Pr[\text{Auth}_6] - \Pr[\text{Auth}_5]\right| \leq \Pr[\text{AskAuth}_6] \leq 2\Pr[\text{AskH}_7] \\ &\Pr[\text{Auth}_7] = \Pr[\text{Auth}_6] = 0\end{aligned}$$

Thus, we have

$$\begin{aligned}\text{Adv}_{\mathcal{P}}^{\text{auth}}(\mathcal{A}) \leq \frac{q_{send}}{|\mathcal{D}|} &+ 5q_h \text{Adv}_{\mathcal{P}}^{\text{CDH}}(t + (q_{send} + q_{exe} + 1) \cdot \tau_e) \\ &+ \frac{q_h^2+6q_{send}}{2^{l+1}} + \frac{(q_{send}+q_{exe})^2}{p}.\end{aligned}$$