The Request for Better Measurement: A Comparative Evaluation of Two-Factor Authentication Schemes

Ding Wang[†], Qianchen Gu[†], Haibo Cheng[†], Ping Wang[†‡]
[†]School of EECS, Peking University, Beijing 100871, China
[‡]National Research Center for Software Engineering, Beijing 100871, China
{wangdingg,qcgu,chenghaibo,pwang}@pku.edu.cn

ABSTRACT

Despite over two decades of continuous efforts, how to design a secure and efficient two-factor authentication scheme remains an open issue. Hundreds of new schemes have wave upon wave been proposed, yet most of them are shortly found unable to achieve some important security goals (e.g., truly two-factor security) and desirable properties (e.g., user anonymity), falling into the unsatisfactory "break-fix-break-fix" cycle. In this vicious cycle, protocol designers often advocate the superiorities of their improved scheme, but do not illustrate (or unconsciously overlooking) the aspects on which their scheme performs poorly.

In this paper, we first use a series of "improved schemes" over Xu et al.'s 2009 scheme as case studies to highlight that, if there are no improved *measurements*, more "improved schemes" generally would not mean more advancements. To figure out why the measurement of existing schemes is invariably insufficient, we further investigate into the state-of-the-art evaluation criteria set (i.e., Madhusudhan-Mittal's set). Besides reporting its ambiguities and redundancies, we propose viable fixes and refinements. To the best of our knowledge, we for the first time demonstrate that there are at least seven different attacking scenarios that may lead to the failure of a scheme in achieving truly two-factor security. Finally, we conduct a large-scale comparative evaluation of 34 representative two-factor schemes, and our results outline the request for better measurement when assessing new schemes.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Authentication

General Terms

Security, Design, Theory, Metric

Keywords

Two-factor authentication; Smart card loss attack; Two-factor security; De-synchronization attack; Measurement.

1. INTRODUCTION

Back to 1991, Chang and Wu [7] proposed the first authentication scheme that combines smart cards and passwords to protect security-critical services such as online banking, e-commerce and e-health. Since then there have been a number of this sort of schemes developed [11,31,38]. Their status is further entrenched by the recent observation [60] that secure and usable leakage-resistance password systems cannot be achieved if we do not resort to trusted devices. The most prominent merit of smart-card-based password authentication schemes (see Fig. 1) is that *two-factor security* can be assured, i.e., only the user who possesses a smart card and knows the correct password can successfully login the server. Thus, they are generally termed "two-factor" schemes [50,61]. A common feature of these early two-factor schemes is that their security largely relies on the tamper-proof property of the smart cards.

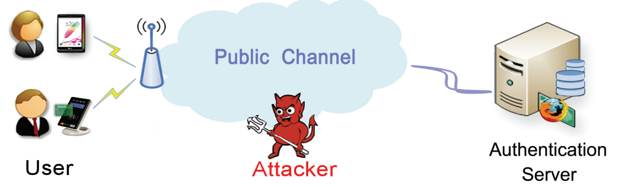


Fig. 1: Smart-card-based password authentication

However, recent research results on side-channel attacks reveal that smart cards, which are traditionally considered as *fully tamper-proof devices*, can no longer be fully trusted once in the hands of a determined attacker. They can be tampered by power analysis [32], reverse engineering techniques [6], and software attacks (launched on software-supported card, e.g., Java Card [27]). This means that, the secret data kept in the card memory can now be extracted when an determined attacker somehow gets access to the smart card. Accordingly, once the smart card factor is breached, these traditional schemes [11, 31, 38] whose security relies on the tamper resistance assumption of smart cards cannot provide truly two-factor security any more.

As there is a constant arms race going on between attackers and security practitioners, even if the physical security of smart cards has been assessed by independent laboratories or certified by third-party authorities (e.g., FIPS-201 [40] and ETSI-TS-102 [13]) at the time of their production, it is highly likely that they will be no longer tamper-proof after a few years of circulation. At FC'13, Zhou et al. [64] provided such a typical example — several versions of commercial GSM SIM cards (with a 16-bit CPU and the COMP128-1 algorithm) are secure against the side-channel attacks in 2002, yet eight years later they are found vulnerable to state-of-the-art side-channel attacks and can be breached in just a few minutes. Considering the long term nature of hardware deployment (because of the cost involved when upgrading), it is important to take adequate preventive measures early in cryptographic developments.

Consequently, it is more admired to design two-factor schemes that rely on the *more realistic assumption* — smart cards can be extracted when acquired (or temporarily acquired) by an attacker. Here is a subtlety to be noted:

this assumption does *not* necessarily suggest smart cards are fully non-tamper-proof devices; Instead, it means that smart cards are only *conditionally non-tamper-proof* — their physical security can be violated only when they are in the hands of the attacker for a sufficiently long period of time (e.g., a few hours [27,32]) for performing the (professional and relatively time-consuming) side-channel attacks. In-depth investigations of this subtlety are referred to [51, 53]. This indicates that, in cases when the user inserts her smart card into a malware-infected card reader, her password may leak yet the sensitive data stored in the card still remain secure, for the user is on the spot and there is no chance for a side-channel attack to be launched.

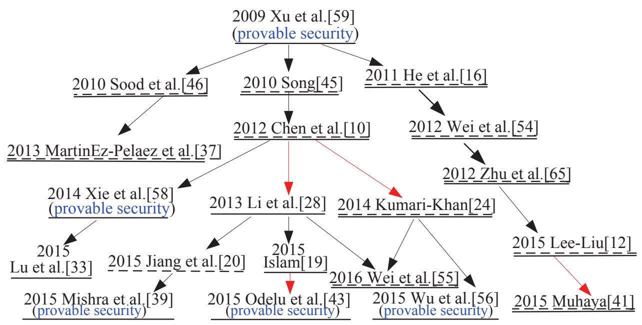


Fig. 2: The evolution of Xu et al.'s 2009 scheme. As we will show in this work, all the 19 schemes underlined by a solid line cannot achieve *truly two-factor security*, while the 11 schemes underlined by a dotted line fail to provide *forward secrecy*.

It is of both theoretical and practical interest to see whether secure and efficient two-factor schemes can be constructed under this more realistic yet challenging assumption about smart cards. A number of attempts [24, 48, 55, 59, 61] have been made. Unfortunately, past research proves that achieving truly two-factor security under this new assumption (i.e., the conditionally non-tamper-resistance assumption about smart cards) is extremely difficult. The pattern of research progress in this area has been of suggested solutions (e.g., [10, 16, 53, 59, 65]), followed by cryptanalysis and improvements (e.g., [28, 41, 45, 48, 54, 55]). Many broken schemes (see one of the most influential one in [59]) are even equipped with a formal proof.

To date we have analyzed over two hundred two-factor authentication schemes, each of which is claimed to be "an improvement" over existing problematic ones. However, whether these "improved versions" indeed improve security is often not well justified. Lots of efforts have been devoted, yet little progress has been made. In this work, we use the improvements over Xu et al.'s seminal scheme [59] as case studies (see Fig. 2) and reveal the unsatisfactory situation of this research pattern. In 2009, Xu et al. [59] for the first time proposed a two-factor scheme with a formal security proof in the random oracle model, and claimed that their scheme can achieve truly two-factor security. However, Song [45] and Sood et al. [46] independently showed that Xu et al.'s scheme cannot fulfill this goal.

A dozen of rebuttals and improvements (e.g., [19, 20, 28, 41, 43] following Xu et al.'s scheme [59] have been presented in succession (see Fig. 2). Among them, four improvements which are representative of certain failures are given by Li et al. in 2013 [28], Kumari-Khan in 2014 [24], Odelu et al. in 2015 [43] and Muhaya in 2015 [41], respectively. Particularly, four improvements (i.e., [39, 43, 56, 58]) are also equipped with a formal proof using the random oracle model, BAN logic and/or π-calculus. In this paper, we show that most of these 19 "improvements" still cannot attain the claimed goal of two-factor security, which is the most crucial goal that a two-factor scheme is designed to achieve. Besides, they are often subject to some other security defects like no forward secrecy and de-synchronization attack.

The unsatisfactory situation regarding Xu et al.'s scheme [59] is by no means an accident. As shown in Fig. 1 of [51], the history of this research area is a monotonous rhythm of "break-fix-break-fix". We believe this is largely due to the insufficient measurement of new and existing schemes. In most cases, the protocol designers present attacks on a previous scheme and propose an improved scheme. Then, they compare it with two or three previous, problematic ones by merely focusing on the dimensions that previous schemes fare poorly, often overlooking the dimensions that previous schemes perform well but the new scheme fails. Invariably, every paper ends up by a conclusion that the new scheme proposed outperforms existing ones. However, the reality is often that the new scheme only achieves some goals and may be no better than the original, problematic one (see Fig. 2 for concrete examples).

We have traced the root cause of the failure in lacking of a sound measurement: the current evaluation criteria are not workable (operable) and hence protocol designers choose to (or have to) use their own customized criteria. In Section 2.3, we show the ambiguities and redundancies in the state-of-the-art criteria set proposed by Madhusudhan and Mittal in 2012 [36]. Besides reporting its deficiencies, we also suggest countermeasures and refinements and test the effectiveness of our suggestions.

The contributions of this paper are three-fold:

- We employ 19 improvements over Xu et al.'s 2009 scheme as case studies to illustrate the lack of fair, thorough measurement of existing schemes in the two-factor authentication research. Our results show that, none of these improved schemes yield improved security and some are even less secure as compared to Xu et al.'s original scheme proposed six years ago. We also identify the fundamental flaws in the reasoning of formal proofs for some "provably secure" schemes.
- We trace the root cause of the current failure by demonstrating that some criteria in the state-of-the-art evaluation set are unworkable due to a number of ambiguities. We further propose viable fixes and refinements. Particularly, for the first time, we show that there are at least eight different smart-card-loss-attack strategies, seven of which would make a scheme unable to achieve truly two-factor security. This not only provides an in-depth understanding of how to measure (two-factor) security but also facilitates protocol designers to be aware of potential threats.
- We provide a large-scale comparative evaluation of 34 representative two-factor schemes based on our refinements of Madhusudhan-Mittal's evaluation set. This provides the missing measurements and thus presents a better understanding of existing schemes. Our measurement results highlight the difficulties in designing a practical two-factor scheme.

2. SYSTEM MODEL, ADVERSARY MODEL AND EVALUATION METRIC

In this section, we briefly sketch the system architecture and adversarial model and reveal the insufficiencies in the state-of-the-art evaluation criteria set that was proposed by Madhusudhan and Mittal in 2012 [36]. As noted in [50], there have been abundant papers dealing with two-factor authentication quite recently (e.g., [10, 22, 29, 47, 48, 53, 62]), yet to the best of knowledge, only a few of them [17, 20, 51, 53] explicate the system architecture and adversarial model which are basic factors in assessing the the security provisions of a scheme. Most ones [42,48,53] do explicate the evaluation metric, yet their metrics are often self-made and far from systematic and mature, though there do exist several evaluation metrics proposed for this use. This outlines the need for an investigation into the reason why these existing metrics are not preferred when protocol designers evaluating their new schemes.

2.1 System architecture

In this work, as with [17, 50], we mainly deal with smart-card-based password authentication for the single server architecture, which is the most general case of two-factor authentication (see Fig. 1). In this sort of schemes, the protocol participants include a set of users and a single authentication server, and typically there are three basic phases, i.e. registration, authentication and password change, and may also be some supplementary phases like re-registration and revocation [42, 57]. In the registration phase, a user U provides her personal data (e.g., identity and transformation of password) to the server S, S personalizes a smart card with some public and sensitive security parameters and issues the card to U. Generally, this phases is conducted only once until the card expires. After completion of this phase, U can login S through the authentication phase. Only the user who can prove that she owns both a valid smart card and the correct password can successfully pass the verification of the server S. Lack of either authentication factor would lead to a login failure. In the password change phase, U can update her password either locally or by interacting with S. More sophisticated schemes may also have additional phases that enable the system to evict a malicious user and revoke a lost card.

2.2 Adversarial model

We adopt the adversarial model that is introduced in [50], and the capabilities of the adversary $\mathcal{A}$ are summarized in Table 1. The capability C-01 means that both the user identity space $\mathcal{D}_{id}$ and the password space $\mathcal{D}_{pw}$ are finite and can be enumerated efficiently. Recent large-scale leakages of user-chosen passwords reveal that $|\mathcal{D}_{pw}|$ is generally about 2^{20} [4,35]. As for user's identity, it is static and often confined to a predefined format and obtained from public sources, and thus it is reasonable to assume that $|\mathcal{D}_{id}| \leq |\mathcal{D}_{pw}| \approx 2^{20} \approx 10^6$.

The capability C-02 is splitted from C-01 to emphasize that when evaluating the *security goals* of a scheme, user identity shall be not considered as a secret value and the security of the system shall not builds on the secrecy of user identity. This, however, does not contradict with the widely hold assumption that the target user's identity is sensitive. When dealing with *the privacy provisions* of a scheme, the target user's identity is sensitive and just what $\mathcal{A}$ attempts to figure out from the publicly available protocol transcripts. What a anonymous two-factor scheme can assure is that, from the public protocol transcripts, $\mathcal{A}$ shall not be able to determine a user's identity. This does not contradicts with the fact that $\mathcal{A}$ determines a user's identity by using non-cryptographic techniques (e.g., social engineering and key logger [63]).

The capability C-1 is the canonical assumption about adversaries in distribution computing. The capability C-2 is the key difference between a security model for two-factor authentication and a security model for password-only authentication. It facilitates to capture the notion of two-factor security and is indeed reasonable according to the recent advances in side-channel attacks [6, 27, 32]. The last two capabilities deal with attacks regarding session key:

Table 1: Capabilities of the adversary

C-01	The adversary $\mathcal{A}$ can enumerate offline all the items in the Cartesian product $\mathcal{D}_{id} * \mathcal{D}_{pw}$ within polynomial time, where $\mathcal{D}_{pw}$ and $\mathcal{D}_{id}$ denote the password space and the identity space, respectively.
C-02	The adversary $\mathcal{A}$ has the capability of somehow learning the victim's identity when evaluating security strength (but not privacy provisions) of the protocol.
C-1	The adversary $\mathcal{A}$ is in full control of the communication channel between the protocol participants.
C-2	The adversary $\mathcal{A}$ may either (i) learn the password of a legitimate user via malicious card reader, or (ii) extract the sensitive parameters in the card memory 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 $\mathcal{A}$ is able to learn the server's long-time private key(s) only when evaluating the resistance to eventual failure of the system (e.g., forward secrecy).

C-3 is used to model know-key attacks, while C-4 is used to capture the notion of forward secrecy.

In all, these six assumptions about adversary capabilities are indeed reasonable and have been little by little accepted (see [14, 17, 36, 42, 53]) since the seminal work of Yang et al. [61]. We note that in most recent schemes (see [18, 28, 41, 42]), the adversarial capabilities C-1 and C-2 have been explicitly stated, while the other four assumptions are invariably implicitly made. In the following sections, our analysis will be based on these six assumptions.

2.3 Evaluation criteria

A concrete, concise and comprehensive evaluation criteria set is essential for a fair assessment of the goodness of existing schemes. A number of criteria sets regarding two-factor authentication have been suggested (e.g., [31, 57, 61]. However, in 2012 Madhusudhan and Mittal [36] demonstrated that these earlier metrics have ambiguities and redundancies, and they presented a new criteria set consisting of nine security requirements (see Table 2) and ten desirable properties (see Table 3) for better assessment of two-factor schemes. It is more comprehensive and workable as compared to existing criteria sets, yet as far as we know, during the past few years it has never been adopted to measure schemes by other researchers.

After careful examination, we find that there are still several ambiguities and redundancies that hinder the effectiveness of Madhusudhan and Mittal's set [36]. Very recently, Wang et al. [50] explicated two ambiguous attributes that dwell in this set: (1) DA1(i.e., no password-related verifier table) shall be splitted into DA1-Weak and DA1-Strong, with the former notion requiring that "no other user-specific data is stored on the server" and the latter notion requiring that "only some non-security-critical user-specific information can be stored on the server"; and (2) DA2 (i.e., freely user password choice) shall be splitted into DA2-Local- Insecure, DA2-Local-Secure and DA2-Interactive, according to where user passwords can be changed *locally* and *securely* (i.e., with verification). They further explicate a subtlety regarding SR6 and other security requirements (e.g., SR2 and SR4): SR6 relates to an attacker who has gained the victim's smart card, while all other security requirements deal with an attacker who is without the victim's smart card. These three explications make Madhusudhan-Mittal's set more concrete and constitute "a first step towards understanding the underlying evaluation metric" [50]. We observe that, there are two other design goals that need to be further explicated, i.e. DA8: user anonymity and SR6: resistance to smart card loss attack.

More specifically, DA8 shall be splitted into DA8-Weak and DA8-Strong, for there are two vastly different notions [29] regarding user anonymity. The basic notion is user

Table 2: Security requirements

SR1	Resistance to DoS attack
SR2	Resistance to impersonation attack
SR3	Resistance to parallel session attack
SR4	Resistance to password guessing attack
SR5	Resistance to replay attack
SR6	Resistance to smart card loss attack;
SR7	Resistance to stolen-verifier attack
SR8	Resistance to reflection attack
SR9	Resistance to insider attack

Table 3: Desirable attributes

DA1	No password-related verifier table
DA2	Freely user password choice
DA3	No password reveal
DA4	Password dependent
DA5	Mutual authentication
DA6	Session key agreement
DA7	Forward secrecy
DA8	User anonymity
DA9	Smart card revocation
DA10	Efficiency for wrong password login

Table 4: A taxonomy of smart-card-loss attacks (PW stands for password)

Smart-card-loss attack types		Need to Extract card	Need to return extracted card	Num of online interactions	Weaknesses exploited	Design goals breached	Typical reference
Passive	Type I	No	Yes	0	No verification when PW change	Usability	Sec. 3.2.2 of [50]
	Type II	Yes	No	0	Definite verifier for PW change	Two-factor security	Sec. 3.2.2 this work
	Type III	Yes	No	0	Partition of non-group PWs	Two-factor security	Sec. 3.5 of [57]
	Type IV	Yes	No	0	Protocol flow 1	Two-factor security	Sec. 3.2.1 this work
	Type V	Yes	Yes	0	Protocol flow 2, pre-computation	Two-factor security, semantic security	Sec. 3.2 and 3.4 of [17]
	Type VI	Yes	Yes	0	Protocol flow 1, pre-computation	Two-factor security	Sec. 3.2.1 of [50]
Hybrid	Type VII	Yes	No	1 with S	Protocol flow 2	Two-factor security	Sec. 5.2.2 this work
	Type VIII	Yes	Yes	1 with U	Protocol flow 3	Two-factor security	Sec. 4.2 of [51]

identity protection, i.e. user identities can not be determined from the protocol transcripts, and some schemes (e.g., [21, 39]) can only achieve this notion of privacy; The advanced notion is user un-traceability, i.e. not only user identities but also user activities can not be determined from the protocol transcripts, and some schemes (e.g., [43, 49]) can achieve this advanced notion of privacy.

The security requirement SR6 shall be splitted into eight different types as listed in Table 4, which is based on our past experience in analyzing over two hundreds of two-factor schemes. One can see that, all these eight types of smart card loss attacks (Type I∼Type VIII) exploit different attacking strategies, and they are quite realistic under the adversary model introduced above. Note that in each type of attack, *at most one* online interactions is involved. This means that most of the attacker's workload are performed *offline*, which is not limited by the security mechanisms of the system (e.g., abnormal login detection, rate-limiting and lockout [12]). Some attacks (e.g., Types I, II and IV) are very generic, while some attacks (e.g, Types III) only work on schemes that imprudently operate user password (or its hash) with a group element. When protocol designers fail to consider any one of these threats, it is highly likely that the proposed scheme will fail to meet SR6. Similarly, when security engineers evaluate schemes, overlooking any of them will lead to unreliable conclusions.

Besides ambiguities, we also note that there are two redundancies regarding SR6 vs. DA4 and SR9 vs. DA3. A scheme that can achieve SR6 means that, $\mathcal{A}$ gaining access to a victim's card should not easily change the password of the smart card, offline guess the password, or impersonate the user. This implies that, in this case, system security relies on user password. Hence, the fulfillment of SR6 indicates that user password shall be dependent. As a result, DA4 is entirely included in SR6. Similarly, DA3 is entirely included in SR9. These two *redundancies* does not impair the practicality of Madhusudhan-Mittal's set as severely as the above *ambiguities* however.

Summary. To the best of knowledge, we for the first provide a taxonomy of smart-card-loss attacks. As far as "the current crux lies in how to achieve truly two-factor security even if the smart cards can be tampered" [51] and seven of the eight types of smart-card-loss attacks in Table 4 can lead to the breach of two-factor security, this in-depth understanding of attacker behaviors constitutes a substantial step forward in resolving "the current crux". By incorporating Wang et al.'s [50] and our above refinements, Madhusudhan-Mittal's set [36] would be much more concrete and practicable, and its effectiveness is tested by a comparative evaluation of 34 representative two-factor authentication schemes in Section 7.

3. REVISITING LI ET AL.'S SCHEME

3.1 Review of Li et al.'s scheme

In this Section, we briefly review Li et al.'s scheme [28]. For ease of presentation, some intuitive notations are listed in Table 5 and will be used through-out this paper. This scheme consists of four phases: registration, login, authentication and password change, and we will follow its descriptions as closely as possible.

Table 5: Notations and abbreviations.

Symbol	Description
U_i	i^{th} user
S	remote authentication server
$\mathcal{A}$	malicious attacker
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
$h(\cdot)$	collision free one-way hash function
$\rightarrow$	a common channel
$\Rightarrow$	a secure channel

Registration phase. Before initializing this phase, the server S chooses two large primes p and q such that $p = 2q + 1$, a master secret key $x \in \mathbb{Z}_q$ and a hash function $h(\cdot) : \{0, 1\}^* \rightarrow \mathbb{Z}_p{}^*$. Whenever U_i enrolls in the server, the following steps will be involved:

(1) U_i chooses her identity ID_i and password PW_i;

(2) $U_i \Rightarrow S : \{ID_i, PW_i\}$.

(3) S computes $A_i = h(ID_i \| PW_i)^{PW_i} \bmod p$ and $B_i = h(ID_i)^{(x+PW_i)} \bmod p$.

(4) $S \Rightarrow U_i$: A card with parameters $\{A_i, B_i, h(\cdot), p, q\}$.

Login phase. When user U_i wants to login to S, the following steps are involved:

(1) U_i inserts her card into a card reader and inputs ID_i and PW_i. The card calculates $A^* = h(ID_i \| PW_i)^{PW_i} \bmod p$,

and rejects the login request if $A_i^* \neq A_i$.

(2) The smart card selects $\alpha \in_R \mathbb{Z}_q^*$ and calculates $C_i = B_i/h(ID_i)^{PW_i} \bmod p$, $D_i = h(ID_i)^\alpha \bmod p$, $M_i = h(ID_i\|C\|D_i\|T_i)$, where T_i is the current timestamp.

(3) $U_i \rightarrow S : \{ID_i, D_i, M_i, T_i\}$.

Authentication phase. Upon receiving the login request from U_i, S and U_i carry out the following steps:

(1) When receiving the login request at time T_i', S verifies that ID_i is valid and that $T_i' - T_i \leq \Delta T$. If either is invalid, the session is terminated.

(2) S computes $C_i'{=}h(ID_i)^x \bmod p$, and $M_i'{=}h(ID_i\|C_i'\|D_i\|T_i)$, and rejects if M_i' is not equal to the received M_i.

(4) S chooses $\beta \in_R \mathbb{Z}_q^*$, reads the current timestamp T_s, and calculates $V_i{=}h(ID_i)^\beta \bmod p$, $sk{=}D_i^\beta \bmod p$ and $M_s{=}h(ID_i\|C_i'\|V_i\|sk\|T_s)$.

(5) $S \rightarrow U_i : \{ID_i, V_i, M_s, T_s\}$.

(6) Upon receiving the response at time T_s', U_i checks the validity of ID_i and whether $T_s' - T_s \leq \Delta T$. If both ID_i and T_s are valid, U_i proceeds.

(7) U_i computes $sk'{=}V_i^\alpha \bmod p$ and $M_s'{=}\ h(ID_i\|C_i\|V_i\|sk'\|T_s)$, and rejects if M_s' is not equal to the received M_s. Otherwise, U_i is assured that S is authentic.

(9) Finally, U_i and S agree a session key $sk = h(ID_i)^{\alpha\beta} \bmod p$ for protecting their ensuing data communications.

3.2 Security analysis of Li et al.'s scheme

Based on the two assumptions C-1 and C-2 as listed in Table 1, Li et al. [28] claimed that their scheme can resist smart card loss attack and achieve two-factor security. Their claim may hold if there only exist honest clients who never deviate from the protocol specifications. Yet, there could be malicious external attackers (as well as dishonest insiders) who often do not stick to the protocol. In the following, we present two kinds of smart-card-loss attacks (see Table 4) on the two-factor security of Li et al.'s scheme under their assumption about $\mathcal{A}$'s capabilities, invalidating their goal of preserving two-factor security.

3.2.1 *Type* IV *attack on two-factor security*

Li et al. [28] showed that, in Chen et al.'s scheme [10], a user's password can be offline guessed once the adversary $\mathcal{A}$ has extracted the secret parameters stored in this user's smart card. Accordingly, Li et al. presented a new scheme to overcome this pitfall. However, precisely the same pitfall still exists in Li et al.'s improvement. The following attack has been shown in [20], yet here we discuss it in much more detail to show its practicality.

In case U_i's smart card is somehow obtained (stolen or picked up) by $\mathcal{A}$, and the stored sensitive information $\{A_i, B_i\}$ can be extracted by some methods under Capability C-1. Further, according to Capability C-2, it is fair to assume that the attacker $\mathcal{A}$ has also intercepted the authentication messages $\{ID_i, D_i, M_i, T_i\}$ exchanged during one normal (successful) login session between U_i and S. Then, $\mathcal{A}$ can obtain U_i's password PW_i as follows:

Step 1. Guesses the value of PW_i to be PW_i^* from the dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $C_i^* = B_i/H(ID_i)^{PW_i^*}$, where B_i is revealed from U_i's smart card and ID_i is intercepted from the public channel;

Step 3. Computes $M_i^* = H(ID_i\|C_i^*\|D_i\|T_i)$, where D_i and T_i is intercepted from the public channel;

Step 4. Checks the correctness of PW_i^* by comparing if the computed M_i^* is equal to the intercepted M_i;

Step 5. Repeats Steps 1, 2, 3 and 4 until the correct value of PW_i is found.

Let $|\mathcal{D}_{pw}|$ denote the size of $\mathcal{D}_{pw}$. The time complexity of the above attack is $\mathcal{O}(|\mathcal{D}_{pw}| * (T_E + T_I + 2T_H))$, where T_E is the running time for modular exponentiation, T_I is the running time for modular inverse operation and T_H is the running time for Hash operation. Thus, the time for $\mathcal{A}$ to determine U_i's password is linear to $|\mathcal{D}_{pw}|$. In reality, the dictionary size is very limited, e.g., $|\mathcal{D}_{pw}| \approx 10^6$ [4, 35], $\mathcal{A}$ may identify the correct password in polynomial time on a common PC. Note that, the limited size of password dictionary is a basic assumption for password-based protocols in the literature (e.g., [2, 61]).

To obtain a concrete running time for our above attacking procedure, we employ the publicly-available cryptographic library MIRACL[1] and Pairing-Based Cryptography (PBC) library[2] to measure the time consumption of related cryptographic operations. We conduct experiments on common PCs or Laptops with varied computation power to simulate the capabilities of a moderate attacker. To be robust, each experiment is repeated for 1000 times. Table 6 summarizes the experimental results. Assuming $|\mathcal{D}| = 10^6$, it follows that the above attacking procedure can be completed within about 45.6($=10^6*$(1.676 ms+1.059 ms+2*0.008 µs) minutes on a common PC.

It is worth noting that the above attacking procedure is conducted *offline* and needs no interaction with the server. After this attack, $\mathcal{A}$ can impersonate U_i at *anytime* until U_i's smart card is revoked by S, because $\mathcal{A}$ now is with both the security parameters of U_i's smart card and the correct password. In other words, there is no way for S to discern the difference between $\mathcal{A}$ and U_i. At ISC'13, Wang-Wang [51] demonstrated an attacking scenario in which $\mathcal{A}$ timely returns back the smart card to U_i after having extracted its parameters, and in this case it is difficult for the victim user U_i to detect the abnormality and ask S to revoke her card. This attacking scenario can also be exploited by our adversary $\mathcal{A}$. All in all, the above attack is indeed practical.

3.2.2 *Type* II *attack on two-factor security*

In the above attack, to succeed, $\mathcal{A}$ needs to obtain both the secret parameters from U_i's smart card and the transcripts exchanged during a successful login session between U_i and S. However, in the following attack, the acquisition of the secret data from U_i's card plus the value of ID_i is sufficient for $\mathcal{A}$ to determine U_i's password. This means $\mathcal{A}$ may even not need to eavesdrop over the channel, because ID_i is often publicly available information (e.g., account number and email address).

Step 1. Guesses the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $A_i^* = H(ID_i\|PW_i^*)^{PW_i^*} \bmod n$, where ID_i is intercepted from the public channel or from other trivial ways;

Step 3. Checks the correctness of PW_i^* by comparing if the computed A_i^* is equal to the extracted A_i;

Step 4. Repeats steps 1, 2 and 3 until the correct value of PW_i is found.

The time complexity of this attack is $\mathcal{O}(|\mathcal{D}_{pw}| * (T_E + T_H))$, which is lower than that of Type IV. According to the timings in Table 2, this attacking procedure can be finished in about 27.9($=10^6*$(1.676 ms+0.008 µs)) minutes.

Remark 1. Both Type II and Type IV attacks demonstrate that, once the smart card factor is compromised, the other factor (i.e., password) can be efficiently breached, and hence the entire system breaks down. This suggests that Li

[1] http://www.shamus.ie/index.php?page=home
[2] https://crypto.stanford.edu/pbc/

Table 6: **Computation evaluation of related cryptographic-primitive operations on common PCs**

Experimental platform (common PCs)	Modular Exp. T_E ($\|n\| = 512$)	Modular Inv. T_I ($\|n\| = 512$)	Point mul. T_P (ECC sect163r1)	Pairing T_{pair} (PBC type A)	MapToPoint T_{m2p} (PBC type A)	Symmetric Enc. T_S(AES)	Hash T_H T_H (SHA-1)	Other lightweight oper.(XOR,$\|$)
Pentium IV 3.06GHz	1.676 ms	1.059 ms	3.107 ms	9.285 ms	185.640 µs	0.312 µs	1.523 µs	0.008 µs
Intel i5-2450M 2.50GHz	0.659 ms	0.504 ms	1.201 ms	4.040 ms	70.827 µs	0.157 µs	0.624 µs	0.007 µs
Intel i7-4790 3.60GHz	0.484 ms	0.353 ms	0.858 ms	2.652 ms	50.916 µs	0.087 µs	0.591 µs	0.006 µs

et al.'s scheme [28], as with its predecessors [10, 45, 59], is still not a truly two-factor scheme under their assumptions and provides no better security with regard to the original ones. Both two types of attacks are also effective in [24, 37].

4. REVISIT KUMARI-KHAN'S SCHEME

4.1 Review of Kumari-Khan's scheme

Here we briefly review the scheme proposed by Kumari-Khan [24], an enhancement over Chen et al.'s scheme [10].

Initialization phase. Notations are used as in Table 5. Server S picks two large primes p and q satisfying that $p = 2q + 1$ and calculates $n = pq$. Then S selects $x \in \mathbb{Z}_q^*$ as its private key and a hash function $h(\cdot)$. S publishes n yet keeps p and q secret. Note $\mathbb{Z}_q^*$ is the multiplicative group of $\mathbb{Z}_q$. S keeps a registration record R_GR to maintain $\{ID_i, T_r\}$, where T_r is the time at registration. To assure confidentiality, S stores $x \cdot p \oplus (ID_i||T_r)$ instead of $ID_i||T_r$.

Registration phase. In this phase, U_i aims to register with the server S:

(1) U_i selects her identity ID_i;

(2) $U_i \Rightarrow S : \{ID_i\}$;

(3) S checks if ID_i is already in R_GR. If not, S creates a registration record R_GR for U_i and stores $x \cdot p \oplus (ID_i \parallel T_r)$ in the backend database.

(4) S calculates $B_i = h(ID_i)^{x+T_r+PW_0}$ $\texttt{mod}$ n, $F_i = (h(ID_i)^{x+T_r}$ $\texttt{mod}$ $n)$ $\oplus PW_0 \oplus ID_i$, and user's encrypted identity $EID_i = E_{x+p}(ID_i \parallel T_r)$. Note that PW_0 is the system's default password; $\{E_{key}(m)$ and $D_{key}(m)\}$ stand for the encryption and decryption of a message m using key, respectively.

(5) $S \Rightarrow U_i : \{B_i, F_i, EID_i, h(\cdot), n, E/D_{key}(\cdot), PW_o\}$.

(6) After receiving the smart card SC, U_i carries out the password change activity to change the default password PW_0 to a new one PW_i.

Login phase. U_i conducts the following steps to login S:

(1) User U_i inserts her SC into a card reader and inputs ID_i and PW_i.

(2) The smart card computes $C_i = (B_i/h(ID_i)^{PW_i})$ $\texttt{mod}$ n, $F_i^* = C_i \oplus PW_i \oplus ID_i$ and checks if $F_i^* = F_i$. If not, the session is rejected.

(3) The smart card selects $\alpha \in_R \mathbb{Z}_n^*$ and computes $D_i = h(ID_i)^{\alpha}$ $\texttt{mod}$ n, $W_i = C_i \cdot D_i$ $\texttt{mod}$ n and $M_i = h(ID_i||C_i||D_i||W_i||T_i)$, where T_u is the current timestamp at the user side.

(4) $U \rightarrow S$: $\{EID_i, D_i, M_i, T_i\}$.

Authentication phase. In this phase, the following procedure is carried out by U_i and S to validate each other as well as calculating a session key.

(1) On receiving the login request $\{EID_i, D_i, M_i, T_i\}$ from U_i, S reads its current timestamp T_{s1} and rejects if $(T_{s1} - T_i) > T$, where T is the allowed maximum transmission delay.

(2) S retrieves $(ID_i||T_r) \leftarrow D_{x+p}(EID_i)$ to get the values $\{ID_i, T_r\}$.

(3) If $(ID_i||T_r)$ exists in R_GR, S computes $C_i^* = h(ID_i)^{x+T_r}$ $\texttt{mod}$ n, $W_i^* = C_i^* D_i$ $\texttt{mod}$ n, and $M_i^* = h(ID_i||C_i^*||D_i||W_i^*||T_i)$.

(4) S checks if $M_i^* = M_i$, and the equivalence means that U_i is legitimate.

(5) S picks the current timestamp T_s and calculates $EID_i^* = E_{x+p}(ID_i||T_r|| T_s)$ and $L_s = E_{C_i^*}(ID_i||EID_i^*||W_i^*||T_s)$. S replaces the value $x \cdot p \oplus (ID_i||T_r)$ with $x \cdot p \oplus (ID_i||T_r||T_s)$ in R_GR. Thus, when U logins next time, S verifies the existence of $(ID_i||T_r||T_s)$ in R_GR to ensure the dynamic nature of $U's$ identity.

(6) $S \rightarrow U_i$: $\{L_s\}$.

(7) Smart card decrypts L_s to get $\{ID_i', EID_i', W_i', T_s'\}$.

(8) Smart card verifies the validity of ID_i' and T_s'.

(9) If ID_i' and T_s' are valid, U_i checks if $W_i^{'} = W_i$ and $EID_i^{'} \neq EID_i$. If not, the session is rejected. Otherwise, smart card updates EID_i with $EID_i^{'}$.

(10) U and S agree the same session key $sk = h(W_i||EID_i^{'}) = h(W_i^*||EID_i^*)$.

Password change phase. U_i can update PW_i to a new one PW_i^{new} as follows:

(1) U inserts card into a card reader and inputs ID_i, PW_i, and PW_i^{new}.

(2) Smart card calculates $C_i = B_i/h(ID_i)^{PW_i}$ $\texttt{mod}$ n, $F_i^* = C_i \oplus PW_i \oplus ID_i$ and verifies if $F_i^* = F_i$. If not, the update request is rejected.

(3) Smart card calculates $B_i^{new} = B_i/h(ID_i)^{PW_i} \cdot h(ID_i)^{PW_i^{new}}$ $\texttt{mod}$ n and $F_i^{new} = C_i \oplus PW_i^{new} \oplus (ID_i)$.

(4) Smart card updates B_i and F_i with B_i^{new} and F_i^{new}.

4.2 Cryptanalysis of Kumari-Khan's scheme

The two assumptions about the adversary's capabilities C-1 and C-2 are also explicitly made in Kumari-Khan's paper [24] when they analyze the security of the scheme of Chen et al. [10]. Naturally, these two assumptions can also be relied on in the following analysis of Kumari-Khan's scheme. We find that this "improved" scheme still fails to achieve the most important goal (i.e., two-factor security) that they claimed. Besides, it can not attain forward secrecy and is prone to de-synchronization attack. Thus, it is actually less secure than Chen et al.'s original scheme.

4.2.1 *Type* IV *attack on two-factor security*

In case U_i's smart card is somehow obtained (stolen or picked up) by $\mathcal{A}$, and the stored data $\{B_i\}$ can be extracted by some means under Capability C-2 (see Table 1). Further, according to Capability C-1, it is fair to assume that $\mathcal{A}$ has also intercepted the authentication messages $\{D_i, M_i, T_i\}$ exchanged during one normal (successful) login session between U_i and the server S. At this point, $\mathcal{A}$ can obtain U_i's password PW_i as follows:

Step 1. Guesses the value of ID_i to be ID_i^* from dictionary space $\mathcal{D}_{id}$ and the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $C_i^* = B_i/H(ID_i^*)^{PW_i^*}$ $\texttt{mod}$ n, where B_i is revealed from U_i's card;

Step 3. Computes $W_i^* = C_i^* \cdot D_i$, where D_i is intercepted from the channel;

Step 4. Computes $M_i^* = H(ID_i||C_i^*||D_i||W_i^*||T_i)$, where D_i and T_i is intercepted from the public channel;

Step 5. Checks the correctness of (ID_i^*, PW_i^*) by comparing if M_i^* is equal to the intercepted M_i;

Step 6. Repeats Step 1, 2, 3 and 4 of this procedure until the correct value of (ID_i^*, PW_i^*) is found.

The time complexity of the above attacking procedure is $\mathcal{O}(|\mathcal{D}_{id}| * |\mathcal{D}_{pw}| * (T_E + T_I + 2T_H))$, where T_E is the running time for modular exponentiation, T_I is the running time for modular inverse operation and T_H is the running time for Hash operation. In reality, the dictionary size is very restricted, e.g., $|\mathcal{D}_{id}| \le |\mathcal{D}_{pw}| \le 10^6$ [4, 35]. Further, according to the timings in Table 6, $\mathcal{A}$ may determine the password in about 24.7 days on a common PC, or spends \$30.56 and costs 16.47 hours by resorting to the Amazon EC2 `C4.4X-large` cloud computing service [1].

Generally, user ID cannot be considered as a secret and actually, it is often publicly available. Thus, there is a high probability for $\mathcal{A}$ to learn the user's ID_i other than guessing it. In this light, the above attack will be more practical. What's more, high performance computers are quite common those days. All this indicates that the above attack is effective even if $\mathcal{A}$ has to figure out both ID_i and PW_i simultaneously.

4.2.2 *Type* II *attack on two-factor security*

Suppose U_i's smart card is somehow obtained (stolen or picked up) by $\mathcal{A}$, and the stored sensitive parameters $\{B_i, F_i\}$ can be extracted by exploiting side-channel attacks under Capability C-2. At this point, $\mathcal{A}$ can obtain U_i's password PW_i as follows:

Step 1. Guesses the value of ID_i to be ID_i^* from dictionary space $\mathcal{D}_{id}$ and the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $C_i^* = B_i / H(ID_i^*)^{PW_i^*} \ \mathtt{mod}\ n$, where B_i is revealed from U_i's smart card;

Step 3. Computes $F_i^* = C_i^* \oplus ID_i^* \oplus PW_i^*$);

Step 4. Checks the correctness of (ID_i^*, PW_i^*) by comparing if F_i^* is equal to the extracted F_i;

Step 5. Repeats Step 1, 2, 3 and 4 of this procedure until the correct value of (ID_i^*, PW_i^*) is found.

The time complexity of this attacking procedure is essentially the same with that of the above Type IV attack.

4.2.3 *No forward secrecy*

What a scheme with forward secrecy can guarantee is that, even if the long-term key(s) of one (or more) participant in the protocol is obtained by $\mathcal{A}$ through leakage or stealing, previously agreed session keys remain confidential. Kumari and Khan [24] explicitly stated that, their new scheme can provide forward secrecy even if the server S's private keys x and p are disclosed. However, in the following we show that this is not the case:

Step 1. Intercepts the message $\{EID_i, D_i, M_i, T_i, L_s\}$ that is exchanged between U_i and the server S during the jth protocol run;

Step 2. Gets ID_i and T_r by decrypting EID_i using S's private keys x and p;

Step 3. Computes $C_i = h(ID_i)^{x+T_i}\ \mathtt{mod}\ n$, $W_i = C_i \cdot D_i\ \mathtt{mod}\ n$, where D_i and T_i are intercepted from the open channel;

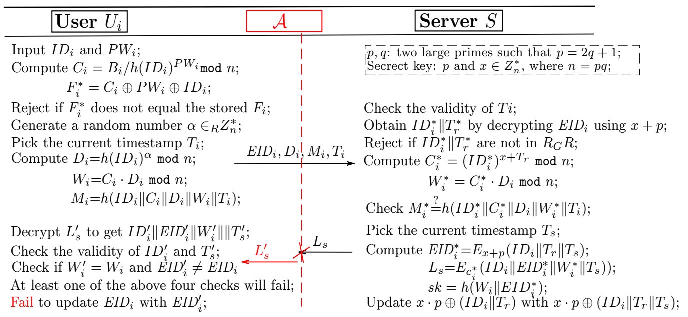
Step 4. Gets EID_i' by decrypting L_s using C_i, where L_s is intercepted from the channel;

Step 5. Computes the session key $sk_j = h(W_i \| EID_i')$;

Our above attack well serves to show that forward secrecy cannot be attained by Kumari-Khan's scheme. The failure of this scheme is essentially due to the violation of "the forward secrecy principle" suggested in [34]: when the scheme's security is based on the discrete logarithmic problem (DLP), at least two exponential operations should be conducted on the server side. Otherwise, the scheme is definitely to be short of forward secrecy.

4.2.4 *De-synchronization attack*

Kumari-Khan's scheme employs a synchronization mechanism to provide the property of user un-traceability. However, this mechanism introduces a serious usability problem. This kind of attack is first highlighted in [50]. In such an attack, the adversary $\mathcal{A}$ can completely destroy the "synchronization" between U_i and S by simply modifying a single protocol transcript, rendering the scheme completely unusable. As summarized in Fig. 3, we show how $\mathcal{A}$ succeeds by altering the second flow from S to U_i with Kumari-Khan's scheme in place.



User U_i

Input ID_i and PW_i;
Compute $C_i = B_i / h(ID_i)^{PW_i} \mathtt{mod}\ n$;
$F_i^* = C_i \oplus PW_i \oplus ID_i$;
Reject if F_i^* does not equal the stored F_i;
Generate a random number $\alpha \in_R Z_n^*$;
Pick the current timestamp T_i;
Compute $D_i = h(ID_i)^{\alpha}\ \mathtt{mod}\ n$;
$W_i = C_i \cdot D_i\ \mathtt{mod}\ n$;
$M_i = h(ID_i \| C_i \| D_i \| W_i \| T_i)$;
Decrypt L_s' to get $ID_i' \| EID_i' \| W_i' \| \| T_s'$;
Check the validity of ID_i' and T_s';
Check if $W_i' = W_i$ and $EID_i' \neq EID_i$
At least one of the above four checks will fail;
Fail to update EID_i with EID_i';

Server S

p, q: two large primes such that $p = 2q+1$;
Secrect key: p and $x \in Z_n^*$, where $n = pq$;
Check the validity of Ti;
Obtain $ID_i^* \| T_r^*$ by decrypting EID_i using $x + p$;
Reject if $ID_i^* \| T_r^*$ are not in $R_G R$;
Compute $C_i^* = (ID_i^*)^{x+T_r}\ \mathtt{mod}\ n$;
$W_i^* = C_i^* \cdot D_i\ \mathtt{mod}\ n$;
Check $M_i^* \overset{?}{=} h(ID_i^* \| C_i^* \| D_i \| W_i^* \| T_i)$;
Pick the current timestamp T_s;
Compute $EID_i^* = E_{x+p}(ID_i \| T_r \| T_s)$;
$L_s = E_{c_i^*}(ID_i \| EID_i^* \| W_i^* \| T_s))$;
$sk = h(W_i \| EID_i^*)$;
Update $x \cdot p \oplus (ID_i \| T_r)$ with $x \cdot p \oplus (ID_i \| T_r \| T_s)$;

Fig. 3: De-synchronization attack

Suppose server S has performed Step 5 of the authentication phase (see Section 4.1) and sends $\{L_s\}$ to U_i as specified, which means S has replaced the value $x \cdot p \oplus (ID_i || T_r)$ with $x \cdot p \oplus (ID_i || T_r || T_s)$ in the backend database $R_G R$. Before $\{L_s\}$ reaches U_i, $\mathcal{A}$ intercepts this message and alters it to $\{X\}$, where X is a randomly selected value. In Steps 8 and 9 of the authentication phase, U_i will find at least one of her four checks will fail. Consequently, U_i will reject S's response and refuse to update EID_i with $EID_i' (= EID_i^*)$ in the card memory. As a result, the consistency of the user-identification data between U_i and S is breached. From now on, U_i will send EID_i to S in her login requests, but S expects EID_i^* and will always reject U_i unless U_i re-registers. One can see that, instead of altering L_s, $\mathcal{A}$ might equally reach her aim by simply dropping L_s. This kind of attack exists in a number of newly proposed schemes [9, 23, 48, 56]. Yet, to the best of our knowledge, there is no easy fix other than radical protocol changes.

5. REVISIT ODELU ET AL.'S SCHEME

In 2015, Odelu et al.'s scheme [43] showed that Islam's scheme [19] (see Fig. 2) is vulnerable to Type II smart-card-loss attack (see Table 4), and they further proposed an improvement. Their new scheme employs three kinds of formal methods to prove its security and is claimed to be "a more secure and robust remote user authenticated key agreement scheme in order to remedy the security flaws" in Islam's scheme [19]. However, as we will show, it is not only vulnerable to Type II attacks, but also introduces a new vulnerability – Type VII smart-card-loss attack.

5.1 Review of Odelu et al.'s scheme

Initialization phase. The server S first chooses a large prime p, a secret key $k_s \in Z_p^*$ and its public key $P_s = g^{k_s}\ \mathtt{mod}\ p$, where $g \in Z_p^*$ is a generator of the cyclic group Z_p^*. Whenever U_i enrolls in S, the following steps are involved:

(1) U_i chooses her identity ID_i;

(2) $U_i \Rightarrow S : \{ID_i\}$.

(3) S checks if $h(ID_i || k_s)$ is found in its database. If yes, S requests U_i to submit a new identity. If not, S computes $C_i = h(ID_i || \ k_s || SID_i)$ and stores $\{C_i, P_s, g, p, h(\cdot)\}$ into the smart card SC_i with identifier SID_i, and S keeps $[h(ID_i \| k_s), SID_i]$ in the database for U_i.

(5) $S \Rightarrow U_i$: A card with data $\{C_i, P_s, g, p, h(\cdot)\}$.

(6) U_i keys her selected password pw_i into the smart card SC_i. Then, SC_i calculates $B_i = C_i \oplus h(pw_i||ID_i)$= $h(ID_i||\ k_s||\ SID_i) \oplus h(pw_i||ID_i)$ and $A_i = h(C_i||\ pw_i||\ ID_i)$. Finally, SC_i replaces C_i by B_i and keeps A_i in its memory. Thus, SC_i finally includes $\{A_i, B_i, P_s, g, p, h(\cdot)\}$.

Login phase. When user U_i wants to login to S, she inserts her smart card into a card reader and keys ID'_i and PW'_i.

(1) SC_i computes $C'_i = B_i \oplus h(pw'_i||ID'_i)$ and $A'_i = h(C'_i||\ pw'_i||\ ID'_i)$, and checks $A'_i \overset{?}{=} A_i$. If not, SC_i rejects. Otherwise, SC_i chooses $\alpha, n_1 \in_R \mathbb{Z}_p^*$, and computes $K_1 = P^{h(\alpha||C_i)}$ $\texttt{mod}$ p, $TID_i = (ID_i||n_1) \oplus h(K_1)$ and $V_i = g^{h(\alpha||C_i)}$ $\texttt{mod}$ p.

(2) $U_i \rightarrow S : \{TID_i, V_i\}$.

Authentication phase Upon receiving the login request from U_i, S and U_i carry out the following steps:

(1) S calculates the decryption key $K_2 = V_{)}^{k_s}i$ $\texttt{mod}$ p and decrypts TID_i using K_2 as $(ID_i||n_1) = TID_i \oplus h(K_2)$. S checks the validity of ID_i by finding if $h(ID_i||k_s)$ is in its database. If there is no entry corresponding to ID_i, S rejects U_i's login request.

(2) S selects $\beta \in_R Z_p^*$ and computes $C_i = h(ID_i||\ k_s||\ SID_i)$, $sk_s = V_i^{h(k_s||\beta)}$ $\texttt{mod}$ p, $V_s = g^{h(k_s||\beta)}$ $\texttt{mod}$ p and $M_s = h(V_i||\ C_i||\ V_s||\ sk_s\ ||n_1)$, where SID_i is retrieved from the backend database corresponding to ID_i.

(3) $S \rightarrow U_i : \{V_s, M_s\}$.

(4) U_i calculates $sk_i = V_s^{H(\alpha||C_i)}$ $\texttt{mod}$ p and checks $M_s \overset{?}{=} h(V_i||\ C_i||\ V_s||\ sk_i||\ n_1)$. If it holds, U_i is assured that S is authentic and computes $M_i = h(sk_i||\ V_s||\ C_i||\ n_1)$. Otherwise, U_i rejects and terminates the session.

(5) $U_i \rightarrow S : \{M_i\}$.

(6) S checks $M_i \overset{?}{=} h(sk_s||\ V_s||\ C_i||\ n_1)$. If yes, S is assured that U_i is authentic and agrees the session key sk_s.

5.2 Cryptanalysis of Odeulu et al.'s scheme

The two assumptions about the adversary's capabilities (i.e., C-1 and C-2) listed in Table 1 are also explicitly made in Odeulu et al.'s paper [58] when they analyze the security of the scheme of Islam [19]. Based on these two assumptions, we show that this scheme is still prone to two kinds of smart card loss attack that breach truly two-factor security. Here we highlight a new attack (i.e., Type VII).

5.2.1 *Type* II *attack on two-factor security*

Odeulu et al. [43] showed that, Islam's scheme [19] cannot provide two-factor security due to a Type II smart-card-loss attack that exploits definite password verifiers. Yet, we show that precisely the same pitfall still exists in [43]. One can see that, there are also definite password verifiers (i.e., A_i and B_i) stored in U_i's card, and thus they can be exploited by $\mathcal{A}$ to offline guess U_i's password.

Step 1. Guesses the value of ID_i to be ID_i^* from dictionary space $\mathcal{D}_{id}$ and the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $C_i^* = B_i \oplus h(pw_i^*||ID'_i)$ and $A_i^* = h(C_i^*||\ pw_i^*||\ ID_i^*)$, where A_i and B_i are extracted from U_i's card.

Step 3. Checks the correctness of (ID_i^*, PW_i^*) by comparing if A_i^* is equal to the extracted A_i;

Step 4. Repeats steps 1, 2 and 3 until the correct value of PW_i is found.

The time complexity of this attack is $\mathcal{O}(|\mathcal{D}_{id}| * |\mathcal{D}_{pw}| * 2T_H)$. According to the timings in Table 6, $\mathcal{A}$ may determine the password in about 4.4 hours on a common PC.

5.2.2 *Type* VII *attack on two-factor security*

In this subsection, we show a new kind of attack against two-factor security. This attack is a hybrid one. More specifically, it first obtains U_i's smart card and extracts the stored data, then it attempts to interact with the server S by impersonating U_i. Using S's response as an oracle, $\mathcal{A}$ can then offline determine U_i's password.

Suppose U_i's smart card is somehow obtained (stolen or picked up) by $\mathcal{A}$, and the stored data $\{A_i, B_i\}$ can be extracted by some means under Capability C-2. Also assume $\mathcal{A}$ has obtained U_i's identity ID_i under the Capability C-02. At this point, $\mathcal{A}$ can obtain U_i's password PW_i as follows:

Step 1. $\mathcal{A}$ chooses $\alpha, n_1, X \in_R \mathbb{Z}_p^*$, and computes $K_1^* = P^{h(\alpha||X)}$ $\texttt{mod}$ p, $TID_i = (ID_i||n_1) \oplus h(K_1^*)$ and $V_i = g^{h(\alpha||X)}$ $\texttt{mod}$ p.

Step 2. $\mathcal{A} \rightarrow S : \{TID_i, V_i\}$.

Step 3. On getting the login request from U_i (actually from $\mathcal{A}$), S will find no abnormality because S can indeed retrieve the correct $ID_i\|n_1$ and find $h(ID_i\|k_s)$ corresponding to ID_i. As a result, S will proceed as usual and send $\{V_s, M_s\}$ to U_i (actually to $\mathcal{A}$).

Step 4. After getting the response $\{V_s, M_s\}$ from S, $\mathcal{A}$ proceeds to the next step;

Step 5. Guesses the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 6. Computes $sk_i = V_s^{h(\alpha||X)}$ and $C_i^* = B_i \oplus h(PW_i^*\ \|ID_i)$, where B_i is revealed from U_i's card;

Step 7. Checks the correctness of PW_i^* by comparing if the computed $h(V_i||\ C_i^*||\ V_s||\ sk_i||\ n_1)$ is equal to the received M_s;

Step 8. Repeats Step 5, 6 and 7 until the correct value of PW_i^* is found.

The time complexity of the above attack is $\mathcal{O}(|\mathcal{D}_{pw}| * (3T_E + 4T_H))$. According to the timings in Table 6, $\mathcal{A}$ may identify the right password in 1.40 hours on a common PC with a 3.06 GHz CPU.

5.2.3 *Flaws in the formal security proofs*

While both Xu et al.'s scheme [59] and Odelu et al.'s scheme [43] are equipped with a formal proof in the random oracle model [3], they are not infallible and actually as insecure as these heuristically analyzed ones (e.g., [20, 28, 41]). *Now a paradox arises*: How can a protocol that was formally proven secure later be found insecure? We provide an answer to this question by showing that their security proofs are flawed. First, we examine the formal proof of Xu et al.'s scheme. Let $\mathcal{A}$ be an attacker against the scheme's semantic security. Their core idea is to employ $\mathcal{A}$ to construct probabilistic polynomial-time (PPT) adversaries for each of the underlying primitives (e.g., Hash and CDH intractability) in such a way that if $\mathcal{A}$ manages to break the semantic security, then at least one of these PPT adversaries succeeds in breaking the security of an underlying primitive. A series of hybrid games $\text{Exp}_n (n = 1, 2 \ldots, 5)$ are defined, starting with the real attack Exp_1 and ending in Exp_5 where $\mathcal{A}$'s advantage is confined to 0, and for which they can bound the difference in $\mathcal{A}$'s advantage between any two consecutive games. This idea is quite routine in provable security [2, 50]. The detailed proof can be found in [59].

In Exp_4, Xu et al. reduces $\mathcal{A}$'s advantage in querying $ID_i\|M\|W\|\text{CDH}(M, W)$ on the hash oracle $h(\cdot)$ to the advantage of $\mathcal{A}$ in solving $\text{CDH}(M, W)$ when given $M = h(ID_i)^m$ and $W = h(ID_{\mathcal{A}})^w$. Unfortunately, this key step fails to consider an *insider* attacker $\mathcal{A}$ that also possesses a valid card and can computes the valid $h(ID_{\mathcal{A}})^x = B - h(PW_{\mathcal{A}})$. Then, $\mathcal{A}$ selects a random nonce $w \in Z_p^*$, chooses any user U_r that $\mathcal{A}$ aims to impersonate, and computes $B' = (h(ID_{\mathcal{A}})^x)^w$. $\mathcal{A}$ sends $\{ID_r, W = h(ID_{\mathcal{A}})^w, C = h(T\|B'\|W\|ID_r), T\}$ to S, where T is the current timestamp. On receiving $\mathcal{A}$'s login request, S will accept it because B'' is computed by S as $B'' = W^x$, and thus $B''(= h(ID_{\mathcal{A}})^x)^w)$ will equal B' and finally the received C will equal $h(T\|B''\|W\|ID_r)$.

Accordingly, in Exp_4 the correct reduction shall be that: $\mathcal{A}$'s advantage in querying $ID_i\|M\|W\|\text{CDH}(h(ID_i)^m, h(ID_i)^w)$ on the hash oracle $h(\cdot)$ is reduced to the advantage of $\mathcal{A}$ in computing $\text{CDH}(h(ID_r)^m, h(ID_r)^w)$ when given the victim's identity ID_r, the value $M = h(ID_r)^m$ and the random number w. In this case, $\mathcal{A}$'s success advantage will be $\Pr[\text{Ext}_4]{=}1$, for it is easy to compute $\text{CDH}(M, W) = M^w \bmod p$.

Now it is time to show the pitfalls in Odelu et al.'s formal proof. After a careful examination, we have to conclude that Odelu et al. misunderstood how to use contradiction in a random oracle model. Essentially, they first suppose that $\mathcal{A}$ can invert hash functions and solve DLP problems, and then they exploit $\mathcal{A}$ to build an adversary that can offline guesses ID_i and PW_i of U_i. They then conclude that, "however, it is a computationally infeasible problem due to the difficulty of solving DLP and inverting the one-way hash function. As a result, it is a contradiction." This kind of proof involves no reasonable reduction and thus can never be seen as a rigorous formal security proof at all. This explains why their scheme fails even if it is equipped with a seemingly formal proof.

The above failures highlight that formal proofs for complex protocols (such as two-factor schemes) are often inevitably intricate and their correctness largely depends on the cryptanalysis experience of the prover (i.e., be aware of all potential threats). This indicates that old-fashioned cryptanalysis continues to play an indispensable role, which highlights the importance of this work.

6. REVISITING MUHAYA'S SCHEME

6.1 Review of Muhaya's scheme

Muhaya's scheme [41] is claimed to be "an improved scheme with session key establishment and user anonymity" to remedy the weaknesses in Zhu's scheme [65], which in turn is an enhanced version over Xu et al.'s 2009 scheme [59]. However, we demonstrate that Muhaya's scheme [41] actually provides *no more* security than Xu et al.'s scheme [59] suggested six years ago.

Registration phase. Before the initialization of this phase, the server S selects two large primes p and q, set $n = pq$. Then, S selects a small prime number e (e.g., $e = 2^{16} + 1$ as generally suggested) as its RSA public key, and computes its RSA private key d such that $ed = 1 \bmod \varphi(n)$. Whenever a user U_i wants to register herself to the medical server S, the following steps are involved:

(1) U_i first chooses her identity ID_i and password PW_i, then picks $N_i \in_R p$, and computes $NPW_i = h(PW_i\|N_i)$.

(2) $U_i \Rightarrow S$: $\{ID_i, NPW_i\}$.

(3) Upon receiving registration request, S calculates $B_i = h(ID_i \oplus d) \oplus NPW_i$ and $C_i = h(ID_i \oplus d) \oplus (ID_i\|NPW_i)$, where d is S's long-term private key;

(4) $S \Rightarrow U_i$: A smart card with data $\{n, e, B_i, C_i, h(\cdot)\}$.

(5) U_i keys N_i into the smart card so that finally the card is with parameters $\{n, e, B_i, C_i, N_i, h(\cdot)\}$.

Login and authentication phase. When user U_i wants to login to S, she inserts her smart card into a card reader and keys ID_i and PW_i.

(1) The smart card computes $NPW_i = h(PW_i||N_i)$ and extracts $B_i^* = B_i \oplus NPW_i$ and $C_i^* = C_i \oplus B_i^*$.

(2) The smart card checks if C_i^* equals B_i^*. If not so, then the smart card terminates the session.

(3) Smart card generates $W_i \in_R p$ and computes $h_i = h(B_i^*||W_i)$ and $X_i = (h_i||W_i||ID_i)^e \bmod n$.

(4) $U_i \rightarrow S$: $\{X_i\}$.

(5) On receiving the login request, S extracts $(h_i^*||W_i^*||ID_i^*) \leftarrow X_i^d \bmod n$ and checks if ID_i^* is valid or not. If not so, S terminates the session.

(6) S checks the equivalence $h_i^* = h(h(ID_i \oplus d)||W_i^*)$ holds or not. If not so, S terminates the session.

(7) S generates $W_{ms} \in_R p$ and computes $h_{ms} = h(ID_i\|W_i^*\|W_{ms})$.

(8) $S \rightarrow U_i$: Response message $\{h_{ms}\|W_{ms}\}$.

The rest part of the scheme has little to do with our discussion and thus it is omitted.

6.2 Security analysis of Muhaya's scheme

The two assumptions about the adversary's capabilities (i.e., C-1 and C-2) listed in Table 1 are implicitly made in Muhaya's work [41] when they analyze the security of the scheme of Zhu [65]. More specifically, the "user impersonation attack" on Zhu's scheme (see Section 3.1 of [41]) implicitly makes the Capability C-1, while the "offline password guessing attack" on Zhu's scheme (see Section 3.2 of [41]) implicitly makes the Capability C-2 — "Suppose U_a (the attacker) steals or finds smart card of U_i and somehow extracts values $\{n, e, ID_i, B_i, N_i\}$ stored inside it." Based on these two assumptions, we show that Muhaya's scheme still cannot provide truly two-factor security and fail to achieve forward secrecy.

6.2.1 *Type* II *attack on two-factor security.*

A scheme with two-factor security can ensure that, only the user possesses a smart card and knows the correct password can successfully login the server [51]. Muhaya [41] showed that Zhu's scheme [65] cannot provide this security goal, because the compromise of the smart card factor leads to the disclosure of the password factor. Yet, we show that precisely the same pitfall still exists in Muhaya's improvement. Suppose U_i's smart card is somehow obtained (stolen or picked up) by $\mathcal{A}$, and the stored data $\{B_i, C_i, N_i\}$ can be extracted by some means under Capability C-2. Now $\mathcal{A}$ can obtain U_i's password PW_i as follows:

Step 1. Guesses the value of ID_i to be ID_i^* from dictionary space $\mathcal{D}_{id}$ and the value of PW_i to be PW_i^* from dictionary space $\mathcal{D}_{pw}$;

Step 2. Computes $NPW_i^* = h(PW_i^*\|N_i)$, where N_i is revealed from U_i's card;

Step 3. Computes $B_i^* = B_i \oplus NPW_i^*$, where B_i is revealed from U_i's card;

Step 4. Computes $C_i^* = B_i^* \oplus (ID_i^*\|NPW_i^*)$;

Step 5. Checks the correctness of (ID_i^*, PW_i^*) by comparing if C_i^* is equal to the intercepted C_i;

Step 6. Repeats Step 1, 2, 3 and 4 of this procedure until the correct value of (ID_i^*, PW_i^*) is found.

The time complexity of the above attacking procedure is $\mathcal{O}(|\mathcal{D}_{id}| * |\mathcal{D}_{pw}| * (2T_H + T_{XOR}))$, where T_H is the running time for Hash operation and T_{XOR} is the running time for XOR operation. In reality, the dictionary size is very restricted, e.g., $|\mathcal{D}_{id}| \leq |\mathcal{D}_{pw}| \leq 10^6$ [4, 35]. Further, according to the timings in Table 6, $\mathcal{A}$ may determine the password in about 3.5 days on a common PC, or spends \$4.32 and costs 2.33 hours by resorting to the Amazon EC2 `C4.4X-large` cloud computing service [1].

6.2.2 *No forward secrecy*

Since two-factor authentication schemes are generally deployed in security-critical applications, forward secrecy is an important property that provides security assurance in case of the ultimate failure of the server (i.e., when S's private key d is somehow leaked). However, in the following we show that this property cannot be achieved:

Step 1. Intercepts the communication messages $\{X_i, h_{ms}, W_{ms}\}$ that is exchanged between U_i and the server S during the jth protocol run;

Step 2. Gets h_i, W_i and ID_i by decrypting X_i using S's private key d;

Step 3. Computes $h_i = h(h(ID_i \oplus d)\|W_i)$;

Step 4. Computes the jth session key $sk_j = h(ID_i\|W_i\| h(ID_i \oplus d)\|h_i\|W_{ms})$;

This shows that the desirable security goal of forward secrecy cannot be attained by Muhaya's scheme. This failure is essentially due to the fact that the scheme makes use of the RSA cryptosystem but violates the principle that "a new temporary RSA key pair must be generated by U_i for each session" [44] to achieve forward secrecy.

Remark 2. We also note that, while Xu et al.'s scheme [59] explicitly claims to provide forward secrecy, Muhaya's scheme [41] as well as its predecessors [16, 54, 65] all do not claim to provide this property. We think that, since forward secrecy is an important property and all the schemes in [16, 41, 54, 65] claim to be an improvement over Xu et al.'s scheme [59], forward secrecy *should have* been considered in the improvements [16, 41, 54, 65]. This well illustrates the undesirable situation in this research area that the protocol designers advocate the superiorities of their scheme, while (perhaps subconsciously) often overlooking the merits that their scheme cannot provide, thus neglecting points on which their scheme performs poorly. This lack of thorough measurement of schemes has led to a long-standing lack of consensus and an unsatisfactory "break-fix-break-fix" cycle.

7. MEASURING IMPROVED SCHEMES

In this section, to test the effectiveness of our refinements to Madhusudhan-Mittal's set [36] and to provide the missing fair, objective comparison of existing schemes, we conduct an evaluation of 34 representative schemes in terms of usability, security and efficiency without hidden agenda. Many them, especially those proposed earlier than 2008, cannot be included here only due to space constraints.

The most critical point one can get from Table 7 is that many newly proposed schemes are less desirable than earlier ones. For instance, all the 19 improvements are inferior to Xu et al.'s original scheme [59] in at least one of the design goals, without regard of efficiency. Among them, Jiang et al.'s scheme [20] is the only can overcome all 8 types of smart card loss attacks, yet it provides no forward secrecy, while this goal is achieved in Xu et al.'s scheme.

No scheme can attain all the design goals DA1-SR9, though one scheme (i.e., [26]) does meet all the security goals SR1-SR9. Fortunately, there is a trend that, as time goes on, schemes are getting better. This is corroborated by the evidence that formal methods lately are gaining considerable popularity in new proposals (see the column right to SR9), though the formal proofs are often immature and flawed (see Sec. 5.2.3) due to the use of insufficient security model (e.g., overlooking some attack types in Table 4).

During our evaluation process, we find that after integrating our refinements, Madhusudhan-Mittal's set [36] is much more concrete and now can be used as an acceptable metric to check which goal is attained or unattained by a scheme under study. Besides the redundancies pointed out in Sec. 2.3, we find some other defects: the security goals regarding session key (i.e., known key attack, unknown key share (UKS) attack, and key control) have not been considered so far. Indeed, a few schemes cannot achieve these important goals: Kim-Kim's [23] and Wang et al.'s [52] schemes are prone to known key attack; Chaudhry et al.'s scheme [9] cannot resist UKS attack; Chen et al.'s scheme [10] is susceptible to key control. Since these goals are essential for measuring an authentication scheme with key agreement, we are considering to re-category, re-word and merge some criteria in the future, in order to include those session-key-related goals while eliminating the identified redundancies.

8. CONCLUSION

In this paper, we first revisited 19 improvements over Xu et al.'s scheme as case studies to show that the current research pattern is unsatisfactory: more "improvements" actually yield less security and there is lack of fair assessment of existing schemes. We then traced the root cause of the failure by showing that, some critical criteria in the state-of-the-art evaluation set are unworkable (impracticable) due to a number of ambiguities. We further propose viable fixes and refinements. The effectiveness of our refinements are tested by a comparative evaluation of 34 representative two-factor schemes. Particularly, we, for the first time, provide a taxonomy of smart-card-loss attacks. This in-depth understanding of attacker behaviors constitutes a substantial step forward in resolving the current crux in achieving truly two-factor security. It is expected that this work would facilitate better measurement of existing and future two-factor schemes.

9. REFERENCES

[1] Amazon elastic compute cloud (Amazon EC2), 2015. `https://aws.amazon.com/ec2/pricing/`.

[2] M. Abdalla, F. Benhamouda, and P. MacKenzie. Security of the j-pake password-authenticated key exchange protocol. In *Proc. IEEE S&P 2015*, pages 571–587. IEEE, 2015.

[3] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *Proc. EUROCRYPT 2000*, pages 139–155.

[4] J. Bonneau. The science of guessing: Analyzing an anonymized corpus of 70 million passwords. In *Proc. IEEE S&P 2012*, pages 538–552.

[5] J. W. Byun. Privacy preserving smartcard-based authentication system with provable security. *Securi. Commun. Netw.*, 2015. Doi:10.1002/sec.1229.

[6] G. Chalupar, S. Peherstorfer, E. Poll, and J. De Ruiter. Automated reverse engineering using lego. In *Proc. Usenix WOOT 2014*, pages 1–10.

[7] C. C. Chang and T. C. Wu. Remote password authentication with smart cards. *IEE Comput. Digital Tech.*, 138(3):165–168, 1991.

[8] Y. Chang, W. Tai, and H. Chang. Untraceable dynamic-identity-based remote user authentication scheme with verifiable password update. *Int. J. Commun. Syst.*, 27(11):3430–3440, 2014.

[9] S. A. Chaudhry, M. S. Farash, and et al. An enhanced privacy preserving remote user authentication scheme with provable security. *Securi. Commun. Netw.*, 2015. Doi:10.1002/sec.1299.

[10] B. Chen and W. Kuo. Robust smart-card-based remote user password authentication scheme. *Int. J. Commun. Syst.*, 27(2):377–389, 2014.

Protocol	Reference	DA1-Strong: *No user data stored*	DA1-Weak: *No password verifier table*	DA2-Local-Secure: *Password friendly*	DA2-Local-Insecure: *Password friendly*	DA2-Interactive: *Password friendly*	DA3: No password reveal	DA4: Password dependent	DA5: Mutual authentication	DA6: Session key agreement	DA7: Forward secrecy	DA8-Strong: *User identity protection*	DA8-Weak: *User un-traceability*	DA9: Smart card revocation	DA10: timely typo detection	SR1: Resist DoS attack	SR2: Resist impersonation attack	SR3: Resist parallel session attack	SR4: Resist pw guessing attack	SR5: Resist replay attack	SR6-*I: Resist card loss attack I*	SR6-*II: Resist card loss attack II*	SR6-*III: Resist card loss attack III*	SR6-*IV: Resist card loss attack IV*	SR6-*V: Resist card loss attack V*	SR6-*VI: Resist card loss attack VI*	SR6-*VII: Resist card loss attack VII*	SR6-*VIII: Resist card loss attack VIII*	SR7: Resist stolen-verifier attack	SR8: Resist reflection attack	SR9: Resist insider attack	**Provable security** in formal model	User side computational cost	Server side computational cost
		Desirable Attributes														Security Requirements																	Protocol Efficiency	
Lu et al.(2016)	[33]	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	$3T_E+4T_H$	$3T_E+3T_H$
Muhaya(2015)	[41]	✓	✓	✓	×	×	✓	✓	✓	✓	×	✓	✓	×	×	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	×	T_E+6T_H	T_E+5T_H
Jiang et al.(2015)	[20]	✓	✓	×	×	✓	×	✓	✓	✓	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	$4T_E+T_S+4T_H$	$2T_E+3T_H$
Wu et al.(2015)	[56]	×	✓	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	$2T_E+2T_S+7T_H$	$2T_E+2T_S+7T_H$
Truong et al.(2015)	[48]	×	✓	✓	×	×	✓	✓	✓	✓	✓	✓	×	✓	×	×	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	×	✓	✓	✓	$2T_C + 5T_H$	$2T_C+6T_H$
Odelu et al.(2015)	[43]	×	✓	✓	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	$3T_E+T_S+4T_H$	$5T_E+T_S+4T_H$
Mishra et al.(2015)	[39]	×	✓	×	×	✓	×	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	$2T_E+6T_H$	$2T_E+5T_H$
Byun(2015)	[5]	×	✓	×	×	✓	×	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	$5T_E+T_S+2T_{m2p}+5T_H$	$5T_E+T_S+3T_H$
Chaudhry et al.(2015)	[9]	✓	✓	✓	×	×	✓	✓	✓	✓	×	×	×	×	✓	×	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	$6T_H$	$2T_S+6T_H$
Xie et al.(2014)	[58]	✓	✓	×	✓	×	×	✓	✓	✓	✓	×	×	×	×	✓	✓	✓	✓	✓	×	✓	✓	×	✓	✓	✓	✓	✓	✓	×	✓	$3T_E+T_S+4T_H$	$5T_E+T_S+4T_H$
Islam(2014)	[19]	×	✓	✓	×	×	✓	✓	✓	✓	✓	×	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	×	×	$3T_E+3T_H$	$2T_E+3T_H$
Kumari et al.(2014)	[25]	✓	✓	✓	×	×	✓	✓	✓	✓	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	×	✓	×	×	$10T_H$	$7T_H$
Tsai et al.(2013)	[49]	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	$T_P + 5T_H$	$3T_P + 5T_H$
Kumari-Khan(2013)	[24]	×	✓	✓	×	×	✓	✓	✓	✓	×	✓	✓	✓	✓	×	✓	✓	✓	✓	×	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	×	$2T_E+T_S+4T_H$	$T_E+2T_S+4T_H$
Li et al.(2013)	[28]	✓	✓	✓	×	×	×	✓	✓	✓	✓	×	×	×	×	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	×	×	$2T_E+2T_I+4T_H$	T_E+4T_H
Lee-Liu(2013)	[26]	×	✓	×	✓	×	✓	✓	✓	✓	×	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	T_E+5T_H	T_E+5T_H
Li-Zhang(2013)	[30]	×	✓	×	×	✓	×	✓	✓	✓	×	✓	✓	✓	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	$3T_P+7T_H$	T_P+7T_H
Chang et al.(2013)	[8]	✓	✓	×	×	✓	×	✓	✓	×	×	×	×	×	×	✓	✓	✓	×	✓	✓	✓	✓	×	×	✓	×	✓	✓	✓	×	✓	$5T_H$	$5T_H$
Kim-Kim(2012)	[23]	×	✓	×	×	✓	×	✓	✓	✓	×	×	×	✓	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	×	×	$2T_E+2T_S+5T_H$	$T_E+2T_S+3T_H$
Chen et al.(2012)	[10]	✓	✓	×	×	✓	×	✓	✓	✓	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	×	×	$2T_E+2T_I+4T_H$	T_E+4T_H
He et al.(2012)	[15]	✓	✓	×	×	✓	✓	✓	✓	✓	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	$2T_E+6T_H$	$2T_E+5T_H$
Wu et al.(2012)	[57]	×	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	$4T_P+1T_{m2p}+5T_H$	$3T_P+5T_H$
Wang(2012)	[53]	×	✓	×	×	✓	×	✓	✓	✓	✓	×	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×	×	$2T_E+T_S+3T_H$	$3T_S+2T_H$
Wei et al.(2012)	[54]	✓	✓	×	✓	×	✓	✓	✓	✓	×	×	×	×	×	✓	✓	✓	✓	✓	×	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	×	$T_E+T_P+5T_H$	$T_E+T_P+5T_H$
Zhu(2012)	[65]	✓	✓	×	✓	×	✓	✓	✓	×	×	×	×	×	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	T_E+5T_H	T_E+4T_H
Wang et al.(2011)	[52]	×	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	$4T_P+T_S+6T_H$	$3T_P+T_S+4T_H$
Khan et al.(2011)	[22]	✓	✓	✓	×	×	✓	✓	✓	✓	×	✓	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	✓	×	$5T_H$	$5T_H$
Li et al.(2010)	[29]	×	✓	×	×	✓	✓	✓	✓	✓	×	✓	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	$3T_S+4T_H$	$3T_P+3T_S+10T_H$
Yeh et al.(2010)	[62]	✓	✓	✓	×	×	×	✓	✓	✓	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	✓	✓	×	✓	$7T_H$	$5T_H$
Song(2010)	[45]	✓	✓	×	×	✓	×	✓	✓	✓	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	×	T_S+4T_H	$T_E+T_S+4T_H$
Sood et al.(2010)	[46]	✓	✓	×	×	✓	×	✓	×	✓	×	×	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	×	×	$3T_E+T_I+3T_H$	$2T_E+T_I+3T_H$
Sun et al.(2009)	[47]	✓	✓	×	✓	×	✓	✓	✓	✓	×	✓	×	×	×	✓	✓	✓	✓	✓	×	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	×	$3T_P+6T_H$	$3T_P+5T_H$
Xu et al.(2009)	[59]	✓	✓	×	×	✓	×	✓	✓	✓	✓	×	×	×	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	$4T_E+4T_H$	$4T_E+4T_H$
Juang et al.(2008)	[21]	×	✓	×	×	✓	✓	✓	✓	✓	×	×	✓	✓	×	×	×	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	×	×	T_S+3T_H	$T_P+2T_S+4T_H$

Note: ✓ means the corresponding design goal is achieved, while × not; More information about the design goals DA1∼SR9 is referred to Sec. 2.3; T_E, T_I,T_P, T_C, T_S and T_H stand for modular exponentiation, modular inversion, ECC point multiplication, phebyshev polynomials, symmetric encryption and hash operation, respectively, and their computational costs on common PCs are referred to Table 6.

[11] M. L. Das. Two-factor user authentication in wireless sensor networks. *IEEE Trans. Wireless Commun.*, 8(3):1086–1090, 2009.

[12] M. Dürmuth, D. Freeman, and B. Biggio. Who are you? A statistical approach to measuring user authenticity. In *Proc. NDSS 2016*, pages 1–15.

[13] ETSI-TS-102: Smart cards; uicc-terminal interface; physical and logical characteristics, Feb. 2010. `http://www.etsi.org/standards`.

[14] F. Hao and D. Clarke. Security analysis of a multi-factor authenticated key exchange protocol. In *Proc. ACNS 2012*, pages 1–11.

[15] D. He, J. Chen, and J. Hu. Improvement on a smart card based password authentication scheme. *J. Internet Tech.*, 13(3):38–42, 2012.

[16] D. He, J. Chen, and R. Zhang. A more secure authentication scheme for telecare medicine systems. *J. Med. Syst.*, 36(3):1989–1995, 2012.

[17] X. Huang, X. Chen, J. Li, Y. Xiang, and L. Xu. Further observations on smart-card-based password-authenticated key agreement in distributed systems. *IEEE Trans. Para. Distrib. Syst.*, 25(7):1767–1775, 2014.

[18] X. Huang, Y. Xiang, E. Bertino, J. Zhou, and L. Xu. Robust multi-factor authentication for fragile communications. *IEEE Trans. Depend. Secur. Comput.*, 11(6):568–581, 2014.

[19] S. Islam. Design and analysis of an improved smartcard-based remote user password authentication scheme. *Int. J. Commun. Syst.*, 2014. Doi:10.1002/dac.2793.

[20] Q. Jiang, J. Ma, G. Li, and X. Li. Improvement of robust smart-card-based password authentication scheme. *Int. J. Commun. Syst.*, 28(2):383–393, 2015.

[21] W.-S. Juang, S.-T. Chen, and H.-T. Liaw. Robust and efficient password-authenticated key agreement using smart cards. *IEEE Trans. Ind. Elec.*, 55(6):2551–2556, 2008.

[22] M. Khan, S. Kim, and K. Alghathbar. Cryptanalysis and security enhancement of a more efficient & secure dynamic id-based remote user authentication scheme'. *Comput. Commun.*, 34(3):305–309, 2011.

[23] K.-K. Kim and M.-H. Kim. An enhanced anonymous authentication and key exchange scheme using smartcard. In *Proc. ICISC 2012*, pages 487–494.

[24] S. Kumari and M. K. Khan. Cryptanalysis and improvement of 'a robust smart-card-based remote user password authentication scheme'. *Int. J. Commun. Syst.*, 27(12):3939–3955, 2014.

[25] S. Kumari, M. K. Khan, and X. Li. An improved remote user authentication scheme with key agreement. *Comput. Electr. Eng.*, 40(6):97–112, 2014.

[26] T. Lee and C. Liu. A secure smart-card based authentication and key agreement scheme for telecare medicine systems. *J. Med. Syst.*, 37(3), 2013.

[27] X. Leroy. Smart card security from a programming language and static analysis perspective, 2013. available at `http://pauillac.inria.fr/~xleroy/talks/language-security-etaps03.pdf`.

[28] X. Li, J. Niu, M. K. Khan, and J. Liao. An enhanced smart card based remote user password authentication scheme. *J. Netw. Comput. Appl.*, 36(5):1365–1371, 2013.

[29] X. Li, W. Qiu, D. Zheng, K. F. Chen, and J. Li.

Anonymity enhancement on robust and efficient password-authenticated key agreement using smart cards. *IEEE Trans. Ind. Elec.*, 57(2):793–800, 2010.

[30] X. Li and Y. Zhang. A simple and robust anonymous two-factor authenticated key exchange protocol. *Secur. Commun. Netw.*, 6(6):711–722, 2013.

[31] I. Liao, C. Lee, and M. Hwang. A password authentication scheme over insecure networks. *J. Comput. Syst. Sci.*, 72(4):727–740, 2006.

[32] J. Liu, Y. Yu, and F.-X. Standaert. Small tweaks do not help: Differential power analysis of milenage implementations in 3G/4G USIM cards. In *Proc. ESORICS 2015*, pages 468–480.

[33] Y. Lu, L. Li, H. Peng, and Y. Yang. Robust anonymous two-factor authenticated key agreement schemem for mobile client-server environment. *Secur. Commun. Netw.*, 2016. Doi: 10.1002/sec.1419.

[34] C.-G. Ma, D. Wang, and S.-D. Zhao. Security flaws in two improved remote user authentication schemes using smart cards. *Int. J. Commun. Syst.*, 27(10):2215–2227, 2014.

[35] J. Ma, W. Yang, M. Luo, and N. Li. A study of probabilistic password models. In *Proc. IEEE S&P 2014*, pages 538–552. IEEE, 2014.

[36] R. Madhusudhan and R. Mittal. Dynamic id-based remote user password authentication schemes using smart cards: A review. *J. Netw. Comput. Appl.*, 35(4):1235–1248, 2012.

[37] R. Martínez-Pelaez, F. Rico-Novella, and J. Forné. Security improvement of two dynamic id-based authentication schemes by sood-sarje-singh. *Elsevier J. Appl. Res. Tech.*, 11(5):755–763, 2013.

[38] N. Mavrogiannopoulos, A. Pashalidis, and B. Preneel. Security implications in kerberos by the introduction of smart cards. In *Proc. ASIACCS 2012*, pages 59–60.

[39] D. Mishra, A. Chaturvedi, and S. Mukhopadhyay. Design of a lightweight two-factor authentication scheme with smart card revocation. *J. Inform. Secur. Appl.*, 23:44–53, 2015.

[40] E. Morse, M. Theofanos, Y. Choong, C. Paul, and A. Zhang. FIPS 201: Personal identity verification of federal employees and contractors. Technical report, NIST, McLean, VA, Aug. 2013. Doi:http://dx.doi.org/10.6028/NIST/.FIPS.201-2.

[41] F. Muhaya. Cryptanalysis and security enhancement of zhu's scheme for telecare medicine system. *Secur. Commun. Netw.*, 8(2):149–158, 2015.

[42] V. Odelu, A. Das, and A. Goswami. A secure biometrics-based multi-server authentication protocol using smart cards. *IEEE Trans. Inform. Foren. Secur.*, 10(9):1953–1966, 2015.

[43] V. Odelu, A. K. Das, and A. Goswami. An effective and robust secure remote user authenticated key agreement scheme using smart cards in wireless communication systems. *Wirel. Pers. Commun.*, 84(4):2571–2598, 2015.

[44] D. Park, C. Boyd, and S.-J. Moon. Forward secrecy and its application to future mobile communications security. In *Proc. PKC 2000*, pages 433–445.

[45] R. Song. Advanced smart card based password authentication protocol. *Comput. Stand. & Inter.*, 32(5):321–325, 2010.

[46] S. Sood and K. Sarje, K.and Singh. An improvement of xu et al.'s authentication scheme using smart cards. In *Proc. Compute 2010*, pages 1–5. ACM, Jan. 2010.

[47] D. Sun, J. Huai, J. Sun, and et al. Improvements of juang et al.'s password-authenticated key agreement scheme using smart cards. *IEEE Trans. Ind. Elec.*, 56(6):2284–2291, 2009.

[48] T.-T. Truong, M.-T. Tran, A.-D. Duong, and I. Echizen. Chaotic chebyshev polynomials based remote user authentication scheme in client-server environment. In *Proc. SEC 2015*, pages 479–494.

[49] J.-L. Tsai, N.-W. Lo, and T.-C. Wu. Novel anonymous authentication scheme using smart cards. *IEEE Trans. Ind. Inform.*, 9(4):2004–2013, 2013.

[50] D. Wang, D. He, P. Wang, and C.-H. Chu. Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment. *IEEE Trans. Depend. Secur. Comput.*, 12(4):428–442, 2015.

[51] D. Wang and P. Wang. Offline dictionary attack on password authentication schemes using smart cards. In *Proc. ISC 2013*, volume 7807 of *LNCS*, pages 1–16.

[52] R. Wang, W. Juang, and C. Lei. Robust authentication and key agreement scheme preserving the privacy of secret key. *Comput. Commun.*, 34(3):274–280, 2011.

[53] Y. G. Wang. Password protected smart card and memory stick authentication against off-line dictionary attacks. In *Proc. SEC 2012*.

[54] J. Wei, X. Hu, and W. Liu. An improved authentication scheme for telecare medicine systems. *J. Med. Syst.*, 36(6):3597–3604, 2012.

[55] J. Wei, W. Liu, and X. Hu. Secure and efficient smart card based remote user password authentication scheme. *Int. J. Netw. Secur.*, 18(4):782–792, 2015.

[56] F. Wu, L. Xu, S. Kumari, and et al. A new authenticated key agreement scheme based on smart cards providing user anonymity with formal proof. *Secur. Commun. Netw.*, 2015. Doi:10.1002/sec.1305.

[57] S. Wu, Y. Zhu, and Q. Pu. Robust smart-cards-based user authentication scheme with user anonymity. *Secur. Commun. Netw.*, 5(2):236–248, 2012.

[58] Q. Xie, N. Dong, D. S. Wong, and B. Hu. Cryptanalysis and security enhancement of a robust two-factor authentication and key agreement protocol. *Int. J. Commun. Syst.*, 2014. Doi:10.1002/dac.2858.

[59] J. Xu, W. Zhu, and D. Feng. An improved smart card based password authentication scheme with provable security. *Comput. Stand. Inter.*, 31(4):723–728, 2009.

[60] Q. Yan, J. Han, Y. Li, and R. H. Deng. On limitations of designing leakage-resilient password systems: Attacks, principles and usability. In *Proc. NDSS 2012*, pages 1–16. The Internet Society, 2012.

[61] G. M. Yang, D. S. Wong, H. X. Wang, and X. T. Deng. Two-factor mutual authentication based on smart cards and passwords. *J. Comput. Syst. Sci.*, 74(7):1160–1172, 2008.

[62] K. H. Yeh, C. Su, and N. W. Lo. Two robust remote user authentication protocols using smart cards. *J. Syst. Soft.*, 83(12):2556–2565, 2010.

[63] F. Zhang, K. Leach, H. Wang, and A. Stavrou. Trustlogin: Securing password-login on commodity operating systems. In *Proc. ASIACCS 2015*, pages 333–344. ACM, 2015.

[64] Y. Zhou, Y. Yu, F. Standaert, and J. Quisquater. On the need of physical security for small embedded devices: a case study with COMP128-1 implementations in SIM cards. In *FC 2013*, pages 230–238.

[65] Z. Zhu. An efficient authentication scheme for telecare medicine information systems. *J. Med. Syst.*, 36(6):3833–3838, 2012.