Hindawi
Security and Communication Networks
Volume 2019, Article ID 2516963, 13 pages
https://doi.org/10.1155/2019/2516963

Review Article

Revisiting Anonymous Two-Factor Authentication Schemes for IoT-Enabled Devices in Cloud Computing Environments

Ping Wang,[1,2,3] **Bin Li**,[1] **Hongjin Shi**,[4] **Yaosheng Shen**,[2] **and Ding Wang**[4]

[1]*School of Software and Microelectronics, Peking University, Beijing 100871, China*
[2]*School of Electronic and Computer Engineering, Peking University Shenzhen Graduate School, Shenzhen, China*
[3]*National Engineering Research Center for Software Engineering, Beijing, China*
[4]*School of EECS, Peking University, Beijing 100871, China*

Correspondence should be addressed to Ding Wang; wangdingg@pku.edu.cn

Received 2 February 2019; Accepted 7 May 2019; Published 23 May 2019

Guest Editor: Fagen Li

Investigating the security pitfalls of cryptographic protocols is crucial to understand how to improve security. At ICCCS'17, Wu and Xu proposed an efficient smart-card-based password authentication scheme for cloud computing environments to cope with the vulnerabilities in Jiang et al.'s scheme. However, we reveal that Wu-Xu's scheme actually is subject to various security flaws, such as offline password guessing attack and replay attack. Besides security, user friendly is also another great concern. In 2017, Roy et al. found that in most previous two-factor schemes a user has to manage different credentials for different services and further suggested a user-friendly scheme which is claimed to be suitable for multiserver architecture and robust against various attacks. In this work, we show that Roy et al.'s scheme fails to achieve truly two-factor security and shows poor scalability. At FGCS'18, Amin et al. pointed out that most of existing two-factor schemes are either insecure or inefficient for mobile devices due to the use of public-key techniques and thus suggested an improved protocol by using only light-weight symmetric key techniques. Almost at the same time, Wei et al. also observed this issue and proposed a new scheme based on symmetric key techniques with formal security proofs in the random oracle model. Nevertheless, we point out that both Amin et al.'s and Wei et al.'s schemes cannot achieve the claimed security goals (including the most crucial goal of "truly two-factor security"). Our results invalidate any use of the scrutinized schemes for cloud computing environments.

1. Introduction

With the emerging paradigm of cloud computing and Internet of Things (IoT), various services are provided over the cloud and accessed by users from IoT-enable devices (e.g., mobile phones and smart watches). As cloud-based services can be accessed anytime and anywhere just with a connection to the Internet and IoT-enable devices are generally of resource-constrained nature, it is important and challenging to protect users and cloud servers from severe security threats, such as fraudulence, eavesdropping, and falsification, posed either by external attackers or malicious internal entities. To guarantee that the resources and services can only be accessed by legitimate parties, user authentication plays an important part in securing electronic transactions by acquiring collaborative evidence. In 2011, Hao et al. [1] suggested the first two-factor authentication scheme that is based on password and smart cards for the cloud computing environments. This initial work has brought about a number of enhanced proposals [2–9] with each different in terms of security [10], anonymity [11], usability [12], and efficiency [13].

Without loss of generality, we consider the most common client-server architecture in which two participants (i.e., a server S and a user U) get involved in two-factor authentication [14]. User U holds a memorable password and a smart card stored with some initial security parameters, and server S only needs to keep some secret key material of the system. Since there is no need to keep a table with password-related verification information on the server side, the server is free from the threat of password dataset leaks and ameliorated from the burden of maintaining a large password dataset.

This feature that makes this type of schemes rather desirable, considering that there are incessant leakages of password databases from large websites[15]. The most important security goal of this kind of schemes is the so-called "two-factor security" [16]. This security concept essentially means that only the user who has the smart card and knows the correct password can be verified by the server.

Most of existing two-factor schemes (e.g, [5, 8, 23]) for cloud computing are built on the basis of generic two-factor schemes like [3, 24]. Nevertheless, past research [20, 25–27] have, again and again, proved that designing a smart-card-based password authentication scheme that can attain "two-factor security" is a tough task. In 2009, Xu et al. [28] developed a smart-card-based password authentication scheme relying on the intractability of computational Diffie-Hellman problem, and stated that their scheme is able to support "two-factor security" under the hypothesis that smart cards can be tampered. In addition, their scheme was "proved secure" in the random oracle model. However, later on Sood et al. [29] found that Xu et al.'s scheme cannot resist against user impersonation attack if the parameters kept in the smart cards can be extracted, invalidating Xu et al.'s claim of ensuring "two-factor security". In 2010, Song [30] independently found this severe flaw in Xu et al.'s scheme. Furthermore, Song presented an improvement to counter the problem emerged in Xu et al.'s scheme.

In 2012, Chen et al. [31] pointed out that various security drawbacks still existed in both Sood et al.'s [29] and Song's [30] schemes. More specifically, Sood et al.'s scheme is susceptible to server impersonation attack and Song's scheme suffers from offline password guessing attack, in case the attacker can obtain those sensitive information kept in the smart card. Chen *et al.* [31] also put forward an improved scheme and argued that their scheme is robust under the condition that the sensitive data in smart card has been revealed by the attacker. It should be noted that recent rapid developments in side-channel attacks have proved that the sensitive information stored in general commercial smart cards could be extracted by power analysis or reverse engineering [32, 33]. Based on a weak yet realistic assumption, Chen et al.'s scheme [31] appears very practical.

However, soon after Chen et al.'s scheme [31] was presented, Ma et al. [20] figured out that it is susceptible to exactly the same problem (i.e., prone to offline password guessing attack and no supply of forward secrecy) with the original scheme (i.e., Song's scheme [30]). Based on their past experience of protocol design and analysis, for the first time Ma et al. [20] suggested three generic principles for designing a secure and efficient two-factor protocol, namely, the public-key principle, the forward secrecy principle, and the security-usability balance principle. Unfortunately, none of the two-factor authentication protocols mentioned above can satisfy all these three design principles and, moreover, as we illustrate, all the schemes studied in this work fail to comply with at least one of these principles.

In 2017, Wu and Xu [17] also observed that previous schemes (e.g., [30, 31]) are vulnerable to various security loopholes (e.g., user impersonation attack and insider attack) and also suggested an enhanced scheme. Wu-Xu argued that their new scheme not only eradicates the security pitfalls being overlooked in previous schemes but also maintains strengths of previous schemes. Notwithstanding their claims, we will show that this scheme still has several serious defects being overlooked: (1) it cannot withstand offline password guessing attack if the data in smart card can be revealed, which means that the primary goal of "truly two-factor security" cannot be satisfying; (2) it is subject to replay attack; (3) it ensures no timely typo detection.

Later on, Roy et al. [19] found that, in Tsai-Lo's scheme [34], a user has to manage different credentials for different services and further suggested a user-friendly scheme which is claimed to be suitable for multiserver architecture and robust against various attacks. Therefore, this protocol shows a good application potential in multiserver cloud computing environments. In this work, we show that Roy et al.'s scheme fails to achieve truly two-factor security and cannot preserve user untraceability. We observe that the first failure of Roy et al.'s scheme is due in large part to the noncompliance with Ma et al.'s [20] security-usability tradeoff principle. Our attacks highlight the necessity of being aware of basic protocol design principles.

In 2018, Amin et al. [21] pointed out that previous schemes (e.g., [35–37]) are vulnerable to various security loopholes (e.g., off-line password guessing attack, insider attack and user impersonation attack) and also developed a new scheme for distributed cloud computing environments. Amin et al. argued that their new scheme not only eradicates the security pitfalls being overlooked in previous schemes but also maintains strengths of previous schemes. Notwithstanding their claims, we will show that this scheme still has several serious defects being overlooked: (1) it cannot withstand offline password guessing attack if the data in smart card can be revealed, which means that the primary goal of "truly two-factor security" cannot be satisfied with; (2) it provides no forward secrecy; (3) it ensures no user untraceability.

Very recently, Wei et al. [23] observed that most of previous schemes (e.g., [36–40]) only provided some heuristic security arguments and little attention has been paid to the formal treatment of protocol security. Unsurprisingly, most of them are vulnerable to various security loopholes. Thus, they suggested an enhanced scheme for cloud computing without using computation-expensive public key operations. They employed a new cryptographic primitive called authenticated encryption scheme which can guarantee both message confidentiality and integrity, and showed that their new scheme can be provably secure in the random oracle model. Though this scheme is armed with a formal proof, we will show that this scheme still has several serious defects being overlooked: (1) it cannot achieve the primary goal of "truly two-factor security"; (2) it provides no forward secrecy; (3) it ensures no user untraceability.

The remainder of the paper is organized as follows: we revisit Wu-Xu's scheme in Section 2; we describe the security loopholes of Roy et al.'s scheme in Section 3; Amin et al.'s scheme is scrutinized in Section 4 and Wei et al.'s scheme is cryptanalyzed in Section 5. Finally, we conclude the paper in Section 6.

Table 1: Notations and abbreviations.

Symbol	Description
U_i	i^{th} user
S	remote server
$\mathcal{M}$	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
$\longrightarrow$	a common channel
$\Longrightarrow$	a secure channel

2. Revisiting Wu-Xu's Scheme

2.1. Review Wu-Xu's Scheme. In this section, we briefly review the chaotic-map based authentication scheme for cloud computing proposed by Wu and Xu [17] in ICCCS 2017. Their scheme consists of four phases: initialization, registration, authentication, and password change. For ease of presentation, we will follow the notations in Wu-Xu's scheme as closely as possible and summarize the notations in Table 1.

2.1.1. Registration Phase. At the very start, the server S generates a random positive integer $s \in Z_p^*$ and a symmetric key cryptosystem with $E_k(.)$ and $D_k(.)$ and then selects a secret key x, and two one-way hash functions $h(.)$ and $h_1(.)$.

Step R1. U_i chooses her identity ID_i, PW_i and b_i, then computes $HPW_i = h(PW_i \| b_i)$

Step R2. $U_i \Longrightarrow S : \{ID_i, HPW_i\}$.

Step R3. S selects a random number r_i, computes $IM_i = E_x((ID_i \oplus r_i) \| r_i)$ and $B_1 = h(ID_i \| HPW_i) \oplus HPW_i \oplus h(x \| IM_i)$, and stores IM_i, B_1, $h(.)$, $E_k(.)/D_k(.)$, s and p into the smart card.

Step R4. $S \Longrightarrow U_i$: A smart card containing $\{IM_i, B_1, E_k(.)/D_k(.), h(.), s, p\}$.

Step R5. U_i computes $B_2 = h(ID_i \| PW_i) \oplus b_i$ and stores B_2 into the card.

2.1.2. Login and Mutual Authentication Phase. When U_i wants to login, the following operations will be performed:

Step 1. U_i inserts the smart card into card reader and inputs ID_i and PW_i.

Step 2. Smart card computes $b_i' = B_2 \oplus h(ID_i \| PW_i)$.

Step 3. Smart card selects two random integers $u \in [1, p + 1]$ and r_u and computes $HPW_i = h(PW_i \| b_i')$, $C_1 = T_u(s)$, $C_2 = B_1 \oplus h(ID_i \| HPW_i)$, $C_3 = h(ID_i \| C_1 \| C_2 \| r_u)$, $C_4 = C_2 \oplus C_3$, $C_5 = E_{C_3}(ID_i \| C_1 \| r_u)$.

Step 4. Smart card$\longrightarrow S : \{C_4, IM_i, C_5\}$.

Step 5. On receiving the message from smart card, S decrypts IM_i and gets ID_i' and r_i' and then computes $C_2' = h(x \| IM_i)$, $C_3' = C_4 \oplus C_2'$.

Step 6. S decrypts C_5 to obtain ID_i'', C_1', and r_u', then checks if $ID_i' \stackrel{?}{=} ID_i''$ and $C_3' \stackrel{?}{=} h(ID_i' \| C_1' \| C_2' \| r_u')$. If all above verifications are correct, S proceeds. Otherwise, the login request is interrupted.

Step 7. S chooses three random integers $v \in [1, p + 1]$, r_s and r_i^{new} and computes $IM_i' = E_x((ID_i' \oplus r_i^{new}) \| r_i^{new})$, $C_6 = T_v(s)$, $sk_s = T_v(C_1')$, $C_7 = h(x \| IM_i')$, $C_8 = h_1(ID_i' \| IM_i' \| C_1' \| C_2' \| C_6 \| C_7 \| sk_s \| r_s)$, $C_9 = C_2' \oplus C_8$, and $C_{10} = E_{C_8}(IM_i' \| C_6 \| C_7 \| r_s)$, where sk_s is server-side session key.

Step 8. $S \longrightarrow U_i : \{C_9, C_{10}\}$.

Step 9. Smart card calculates $C_8' = C_9 \oplus C_2$ and obtains IM_i'', C_6', C_7', and r_s' by decrypting C_{10}. Smart card further computes $sk_u = T_u(C_6')$ and checks $C_8' \stackrel{?}{=} h_1(ID_i \| IM_i'' \| C_1 \| C_2 \| C_6' \| C_7' \| sk_u \| r_s')$. If they are not equal, U_i aborts the communication. Otherwise, smart card computes $B_1' = C_2 \oplus C_7' \oplus B_1$ and replaces (B_1, IM_i) with (B_1', IM_i'').

2.2. Cryptanalysis of Wu-Xu's Scheme. In this section, the security loopholes of Wu-Xu's scheme [17] will be pointed out. More specifically, it is prone to offline password guessing attack and suffers from replay attack, which makes this scheme unpractical for real use. Before giving the detailed security analysis, we first define the various adversary models for smart-card-based password authentication.

2.2.1. Adversary Models. To analyze the security provisions of password-based authentication schemes using smart cards, generally three assumptions about the attacker's capabilities are made since the landmark work of Yang et al. [24], and we summarize them as follows:

Assumption 1. The malicious attacker $\mathcal{M}$ is able to eavesdrop, delete, insert, modify, or block any transcripts communicated in the public channel. That is to say, the communication channel between the common users and the server can be completely manipulated by $\mathcal{M}$. This well complies with the standard adversary model that is widely accepted for distributed computing [41].

Assumption 2. The malicious attacker $\mathcal{M}$ can somehow get the victim user's smart card and use side-channel attack techniques to acquire sensitive security parameters from the card memory, which is reasonable according to the recent research developments in side-channel attacks [32, 33].

Assumption 3. User's password space is very constrained and the malicious attacker $\mathcal{M}$ can offline enumerate it. To be user-friendly, most protocols (e.g., the ones in [3, 24, 42–44]) enable the users to select their own password at will in the initial process of registration or later process of password change. Because human beings are incapable of memorizing random strings, instead they are likely to choose passwords that relate to their personal lives or short strings for convenience. As a result, these human-generated passwords are often very weak and belong to a small dictionary [45–47].

Note that the nontamper resistant assumption about smart cards are conditional, i.e., under the conditions that (1) smart cards have somehow been in the possession of the attacker for a relatively long time (e.g., at least a few days), which should be sufficient for launching a side-channel attack; (2) the user is not on the scene; otherwise the user will observe the attack, for side-channel attacks generally need special instruments and professional/particular operations. This is why smart cards are superior to memory sticks, even though nontamper assumption about smart cards has been made: memory sticks are nontamper resistant unconditional. See more explications in [48].

Obviously, if both *Assumptions* 2 and 3 hold simultaneously, then an attacker (without any other assumptions/abilities) can successfully impersonate a victim user and any scheme is trivially insecure. Therefore, it is widely regarded that attackers should not be granted to obtain a victim user's smart card as well as his password [3, 24, 49].

In [17], Wu and Xu reported that their scheme is secure under the above three assumptions. For example, they stated that their scheme can withstand offline password guessing attacks even if the security parameters in smart card have been extracted. However, contrary to their claims, we will illustrate that this scheme is still vulnerable to offline guessing as well as other pitfalls. Based on the above listed assumptions, we cryptanalyze the security provisions of Wu-Xu's scheme in the following and assume that $\mathcal{M}$ can extract the secret data $\{B_1, B_2, h(\cdot), E_k(\cdot), D_k(\cdot)\}$ stored in U_i's smart card, and $\mathcal{M}$ can also eavesdrop the messages $\{C_4, IM_i, C_5\}$ exchanged between the parties.

2.2.2. Offline Password Guessing Attack. It is known that password-based authentication schemes are apt to be subject to two kinds of attacks regarding guessing [16, 25], i.e., offline password guessing and online password guessing, due to the limited size of the password space. Among them, the online guessing is relatively easy to detect due to the abnormal, large number of login requests issued by the attacker against the victim account within a short duration, and thus it can be countered by rate-limiting [50].

It is worth noting that when targeted online password guessing [47] is considered, attackers will no longer need to launch a large number of login attempts to try candidate passwords. Instead, targeted online password guessing attacks are rather effective and, as reported by Wang et al. [47], the attacker $\mathcal{M}$ can have an over 20% of success rate within 100 login attempts against normal users, if $\mathcal{M}$ has obtained some personal information (e.g., name, birthday) of the victim user. As a quick response to the results of [47], the US Digital identity guideline NIST 800-63B [51] has been revised and the following sentence was removed: "online guessing can be readily addressed by throttling the rate of login attempts permitted" (see more details in [52, 53]). Further, NIST 800-63B proposed four countermeasures such as CAPTCHA, exponential time delay, and risk-based authentication (see Section 5.2.2 of [51]).

In contrast, offline password guessing attack cannot be easily detected. In this attack, the attacker $\mathcal{M}$ first attempts to look for some pieces of information that can be exploited as the comparison target of his password guesses and then locally determines the exactly correct password by repeatedly testing all the candidates. Since this attack is executed without online communication with the server, there is no means for the server to detect and thwart. In all, no matter in trawling or targeted online guessing, $\mathcal{M}$ needs to interact with the server and there is the possibility that $\mathcal{M}$ may be detected, but in offline password guessing there is no such possibility. Consequently, offline password guessing is a more serious threat if password-related information has been leaked.

Wu and Xu [17] claimed that an attacker is unable to, in an offline manner, determine a user's password even if the sensitive data B_1 has been revealed from user's smart card. However, the following attacking procedure illustrates that this claim is not tenable. Suppose that U_i's smart card is lost/stolen and the attacker $\mathcal{M}$ obtains it. Then $\mathcal{M}$ extracts the content $\{B_1, B_2, h(\cdot)\}$ by using the methods introduced in [33]. The following procedure describes our proposed offline password guessing attack:

Step 1. $\mathcal{M}$ choose a pair (ID_i^*, PW_i^*) from the identity space $\mathcal{D}_{ID}$ and dictionary space $\mathcal{D}_{PW}$, respectively.

Step 2. $\mathcal{M}$ computes $b_i^* = B_2 \oplus h(ID_i^* \parallel PW_i^*)$, where B_2 is revealed from U_i's smart card.

Step 3. $\mathcal{M}$ computes $C_2^* = B_1 \oplus h(ID_i^* \parallel h(PW_i^* \parallel b_i^*)) \oplus h(PW_i^* \parallel b_i^*)$, where B_1 is recovered from U_i's smart card.

Step 4. $\mathcal{M}$ calculates $C_3^* = C_2^* \oplus C_4$, where C_4 is eavesdropped from the open channel.

Step 5. $\mathcal{M}$ decrypts C_5 by using C_3^* to obtain $ID_i' \oplus C_1' \oplus r_u'$, where C_5 is eavesdropped from the open channel.

Step 6. $\mathcal{M}$ calculates $C_3^* = h(ID_i^* \parallel C_1' \parallel C_2^* \parallel r_u')$.

Step 7. $\mathcal{M}$ examines the authenticity of (ID_i^*, PW_i^*) pair by verifying if $C_4 \stackrel{?}{=} C_2^* \oplus C_3^*$.

Step 8. $\mathcal{M}$ returns to Step 1 of this procedure until the right pair of (ID_i, PW_i) is obtained or all pairs in $\mathcal{D}_{ID} \times \mathcal{D}_{PW}$ are exhausted.

It is obvious that the above procedure is with a time complexity of $\{\mathcal{O}(|\mathcal{D}_{ID}| * |\mathcal{D}_{PW}|)\} * (T_D + 4T_H + 7T_X)$, where T_E, T_H, and T_X denote the execution time of modular exponentiation, hash, and XOR operation, respectively. Based on the results reported in [16, 54], the offline password

guessing attack is able to be carried out in seconds on a common computer, for in practice the size of identity space $\mathscr{D}_{ID}$ and the size of dictionary space $\mathscr{D}_{PW}$ are rather limited and $\mathscr{M}$ could try all the possible passwords through an offline method [20, 49]. All in all, an type II attacker $\mathscr{M}$ can guess (ID_i, PW_i) within polynomial time bound; it follows that our suggested attack is indeed effective.

2.2.3. Replay Attack. Resistance to replay attack is a very basic security goal of any cryptographic protocol [16, 24]. However, Wu-Xu's scheme fails to achieve this goal. More specifically, Wu-Xu's scheme employs random numbers but not timestamps to achieve the freshness of messages. Yet, this scheme has only two protocol flows, making it inherently vulnerable to replay attack. As is well known, any two-flow random number based scheme is unable to achieve explicit authentication while resisting replay attack, because $\mathscr{M}$ can simply replay the first message of a successful protocol run to impersonate the legitimate user, and the server can never know whether the replayed message is fresh or not unless the server maintains a table of all received messages. However, maintaining a table of all received messages is practically undesirable. In all, replay attack is quite realistic against Wu-Xu's scheme.

3. Revisiting of Roy Et Al.'s Scheme

3.1. Review of Roy Et Al.'s Scheme. In this section, we first concisely review Roy et al.'s scheme [19] proposed in 2017. This scheme is an improvement over Tsai-Lo's scheme [34] and aims to attain forward secrecy that is lacked in Tsai-Lo's scheme. Roy et al.'s protocol involves three participants, i.e., the mobile user (MU_i), the cloud server (CS_j), and registration center (RC). There are five phases in their scheme: registration, login, authentication and key establishment, password change, and revocation of mobile device. The notations and initial system parameters employed in Roy et al.'s scheme are same as employed in the scheme of Wu-Xu (see Table 1).

3.1.1. Mobile User Registration

Step 1. MU_i chooses her identity ID_i, password PW_i, biometrics B_i, and two 128-bit random numbers b and k.

Step 2. MU_i produces $(\theta_i, \phi_i) = Gen(B_i)$ and computes the masked password $RPWB_i = H(ID_i \parallel H(PW_i \parallel \theta_i \parallel b))$.

Step 3. $MU_i \Longrightarrow RC : \{ID_i, (RPWB_i \oplus k)\}$.

Step 4. RC selects an 1024-bit master secret key X_j for server CS_j. RC also selects an 1024-bit random number r_{ij} for each MU_i and CS_j pair.

Step 5. RC computes $A_{ij} = H(H(ID_i \oplus r_{ij}) \parallel X_j)$, $V_{ij} = A_{ij} \oplus RPWB_i$, and $RID_{S_j} = H(ID_{S_j} \parallel X_j)$ as the pseudoidentity of CS_j.

Step 6. RC selects a unique and random temporary identity TID_i for MU_i and saves n server key-plus-id combinations $\{TID_i, (ID_{S_j}, V_{ij}, RID_{S_j}) \mid 1 \le j \le n\}$ in mobile device of MU_i.

Step 7. $RC \Longrightarrow MU_i$: A mobile device contains $\{TID_i, (IDS_j, V_{ij}, RID_{S_j}) \mid 1 \le j \le n\}$.

Step 8. MU_i computes $D_i^1 = H(PW_i \parallel \theta_i) \oplus b$, $D_i^2 = H(ID_i \parallel PW_i \parallel \theta_i \parallel b)$, $V_{ij}' = V_{ij} \oplus k = A_{ij} \oplus H(ID_i \parallel H(PW_i \parallel \theta_i \parallel b))$, $RID_{ij} = TID_i \oplus H(ID_i \parallel V_{ij}')$, and $RID_{S_j}' = RID_{S_j} \oplus H(\theta_i \parallel b)$ for $1 \le j \le n$.

Finally, MU_i stores ϕ_i, D_i^1, D_i^2, V_{ij}'s, RID_{ij}s, and RID_{S_j}'s into her own mobile device, and deletes V_{ij}s, TID_i, and RID_{S_j}s from the mobile device.

3.1.2. Cloud Server Registration

Step 1. CS_j chooses her identity ID_{S_j}.

Step 2. $CS_j \Longrightarrow RC : \{ID_{S_j}\}$.

Step 3. RC provides the master secret key X_j to each CS_j.

Step 4. For all MU_is, the RC saves the credentials $\{TID_i, (ID_i, r_{ij})\}$ (for $1 \le i \le m$) in database of CS_j, and also stores $\{ID_{S_j}, X_j\}$ in the database of CS_j.

Finally, RC saves pair (ID_i, SN_i) in its own database, where SN_i is the serial number of MU_i's mobile device.

3.1.3. Login Phase. When MU_i wants to login to CS_j, the following operations shall be executed:

Step L1. MU_i inputs her identity ID_i, password PW_i, and biometrics B_i' into her own mobile device. MU_i computes $\theta_i = Rep(B_i', \phi_i)$ with ϕ_i through the fuzzy extractor reproduction procedure and generates $b' = D_i^1 \oplus H(PW_i \parallel \theta_i)$ with the stored parameter D_i^1, MU_i.

Step L2. MU_i computes $H(ID_i \parallel PW_i \parallel \theta_i \parallel b')$ and checks whether $D_i^2 = H(ID_i \parallel PW_i \parallel \theta_i \parallel b')$. An equality indicates that MU_i is legal.

Step L3. MU_i calculates $RPWB_i = H(ID_i \parallel H(PW_i \parallel \theta_i \parallel b'))$. MU_i also generates $A_{ij} = V_{ij}' \oplus RPBW_i$ using the device parameter V_{ij}'.

Step L4. MU_i selects an 128 bit random number RN_i, generates the current timestamp TS_i, and then computes:

$C_1 = A_{ij} \oplus RN_i \oplus TS_i \oplus H(IDS_j)$, $H_1 = H(ID_i \parallel C_1 \parallel RN_i \parallel TS_i)$, $TID_i = RID_{ij} \oplus H(ID_i \parallel V_{ij}')$, $RID_{S_j} = RID_{S_j}' \oplus H(\theta_i \parallel b')$, and $TID_i^* = TID_i \oplus H(RID_{S_j} \parallel TS_i)$.

Step L5. $MU_i \longrightarrow CS_j : \{TID_i^*, C_1, H_1, TS_i\}$.

3.1.4. Authentication Phase. In this phase, CS_j and MU_i mutually authenticate each other and establish a session key.

Since this phase is unrelated to our discussions, we omit it.

3.2. Flaws in Roy Et Al.'s Scheme. Recall that the three assumptions listed in Section 3 are also clearly made in Roy et al.'s scheme. However, we observe that this scheme still remains feasible for an attacker to offline guess a user's password. This means that the primary goal of "truly two-factor security" cannot be satisfied. In addition, the scheme cannot provide forward secrecy and sound scalability.

3.2.1. Offline Password Guessing Attack. Noting that Roy et al.'s scheme [19] is originally a three-factor one, here we are only interested in its two-factor version by assuming that the third factor (i.e., the biometric) has been known to $\mathscr{A}$. This is realistic as users' biometrics are constant during their lives, and how to protect user biometric template is still an open issue [55]. We find that this scheme cannot achieve truly two-factor security: it is subject to two types of offline password guessing attack. This in turn indicates that it cannot achieve truly three-factor security.

Type-I Attack. Suppose U_i's biometric B_i and the secret parameters $\{D_i^1, D_i^2, \varphi_i, h(\cdot)\}$ stored in the smart card are somehow obtained by $\mathscr{A}$. At this point, $\mathscr{A}$ can find out U_i's identity and password as follows:

Step 1. Guesses U_i's identity ID_i^* and password PW_i^* from dictionary space $\mathscr{D}_{id}$ and $\mathscr{D}_{pw}$.

Step 2. Computes $\theta_i = Rep(B_i, \varphi_i)$, $b^* = H(PW_i \parallel \theta_i) \oplus D_i^1$, where D_i^1 is extracted from the smart card.

Step 3. Computes $D_i^{2*} = H(ID_i^* \parallel PW_i^* \parallel \theta_i \parallel b^*)$, where θ_i is extracted from the smart card.

Step 4. Checks the validity of (ID_i^*, PW_i^*) by comparing the calculated D_i^{2*} with the extracted D_i^2.

Step 5. Repeats Steps 1~4 until finds the correct pair of (ID_i^*, PW_i^*).

The time complexity of this attack is $\mathcal{O}(|\mathscr{D}_{id}| * |\mathscr{D}_{pw}| * 2T_H + T_B)$ [3, 16]. Generally, it is only needed to calculate the biohashing function once; thus T_B can be ignored in practice. According to the running time in [16], $\mathscr{A}$ may complete the above attacking procedure within 17.6 days on a Laptop. This issue arises due to the inherent "usability-security tension": to achieve local password change (i.e., C-2 in [3]) and timely typo detection (i.e., C-9 in [3]); there is an explicit password verifier $D_i^2 = H(ID_i \parallel PW_i \parallel \theta_i \parallel b)$ stored in U_i's smart card, yet this verifier leads to a Type-I offline password attack.

To eliminate this security issue without loss of usability, a promising countermeasure is to adopt the "fuzzy-verifier" technique [20] and store $D_i^2 = h((H(ID_i \parallel PW_i \parallel \theta_i \parallel b)) \bmod n)$ in U_i's smart card, where n specifies/confines the capacity of (ID, PW) pair, $2^4 \le n \le 2^8$. As a result, even if $\mathscr{A}$ gets D_i^2, she can not figure out the correct (ID, PW) from the above attack, because there will be $|\mathscr{D}_{id} * \mathscr{D}_{pw}|/n \approx 2^{32}$ candidate (ID, PW) pairs that make $D_i^{2*} = D_i^2$ in Step 4. To further identify the exactly correct (ID, PW) pair, $\mathscr{A}$ needs to interact with the server, and we can adopt the "honeywords" technique [3, 56] to confine $\mathscr{A}$'s advantage to a very limited value.

Type-II Attack. In the above attack, $\mathscr{A}$ does not need the protocol messages. In this attack, we assume that $\mathscr{A}$ can somehow obtain user's smart card and extract its secret parameters $\{D_i^1, D_i^2, V_{ij}', \varphi_i, h(\cdot)\}$ and also can eavesdrop the login messages $\{TID_i^*, C_1, H_1, TS_i\}$ from the public channel. Now, $\mathscr{A}$ can *offline* guess U_i's password and identity simultaneously as follows:

Step 1. Guesses the value of ID_i^*, PW_i^* from dictionary space $\mathscr{D}_{id}$ and $\mathscr{D}_{pw}$.

Step 2. Computes $\theta_i = Rep(B_i, \varphi_i)$, $b^* = H(PW_i \parallel \theta_i) \oplus D_i^1$, where D_i^1 is extracted from the smart card.

Step 3. Computes $A_{ij}^* = V_{ij}' \oplus H(ID_i^* \parallel H(PW_i^* \parallel \theta_i \parallel b^*))$.

Step 4. Computes $RN_i^* = C_1 \oplus A_{ij}^* \oplus TS_i \oplus H(ID_{S_j})$.

Step 5. Computes $H_1^* = H(ID_i^* \parallel C_1 \parallel RN_i^* \parallel TS_i)$.

Step 6. Checks the correctness of (ID_i^*, PW_i^*) by comparing if the computed H_1^* equals the intercepted H_1.

Step 7. Repeats Steps 1~6 of the above procedure until find the correct value of (ID_i^*, PW_i^*).

The time complexity of the above attacking procedure is $\mathcal{O}(|\mathscr{D}_{id}| * |\mathscr{D}_{pw}| * (5T_H + T_B))$, and $\mathscr{A}$ may complete the procedure within 44 days on a Laptop. In comparison, the Type-I attack is more practical.

3.2.2. No Forward Secrecy. A scheme that supports forward secrecy ensures that even after the long-term private key(s) of one or more participants were leaked, previously agreed session keys remain secure [27]. This is important, especially when considering the serious situations of today's clouds like the compromise of cloud servers (e.g., [15]).

If an attacker $\mathscr{M}$ has obtained the server CS_j's long-term key X_j from the breached server S and intercepted the messages $\{TID_i^*, C_1, H_1, TS_i\}$ that are exchanged between U_i and S's authentication process from the public channel. $\mathscr{M}$ is able to figure out the session key using the following method:

Step 1. $\mathscr{M}$ computes $RID_{S_j} = H(ID_{S_j} \parallel X_j)$ and $TID_i = TID_i^* \oplus H(RID_{S_j} \parallel TS_i)$

Step 2. $\mathscr{M}$ computes $A_{ji} = H(H(ID_i \oplus r_{ij}) \parallel X_j)$, $RN_j = C_1 \oplus TS_i \oplus H(ID_{S_j}) \oplus A_{ji}$, $M_1 = C_1 \oplus TS_i \oplus H(ID_{S_j}) \oplus B_{ji}$;

Step 3. $\mathscr{M}$ calculates the session key $sk = H(ID_i \parallel ID_{S_j} \parallel A_{ji} \parallel M_1 \parallel RN_j \parallel TS_i \parallel TS_j)$.

With the session key sk computed, the entire session will be no secret to $\mathcal{M}$.

3.2.3. Poor Scalability. In Roy et al.'s scheme, the user side stores all the cloud servers (i.e., CS_j, $1 \le j \le n$) related information: $RID_{ij} = TID_i \oplus H(ID_i \parallel V'_{ij})$ and $RID'_{S_j} = RID_{S_j} \oplus H(\theta_i \parallel b)$ for $1 \le j \le n$. This means that when a new server arrives, all the users have to reregister with the registration center RC. This shows poor scalability. Similarly, the server side stores all the users (i.e., U_i, $1 \le i \le m$) related information: the credentials $\{TID_i, (ID_i, r_{ij})\}$ (for $1 \le i \le m$) in database of CS_j. This means that when a new user arrives, all the servers have to reregister with the registration center RC.

4. Revisiting Amin Et Al.'s Scheme

In 2018, Amin et al. [21] pointed out that previous schemes (e.g., [35–37]) are vulnerable to various security loopholes and developed a new scheme for distributed cloud computing environments. However, here we show that this scheme still has several serious defects being overlooked (i.e., offline password guessing attack, no forward secrecy, and no user untraceability).

4.1. Review Amin Et Al.'s Scheme

4.1.1. Cloud Server Registration Phase. The registration phase is partitioned into two parts: cloud server registration and user registration. When the cloud server S_m registers, S_m chooses her identity SID_m, a random nonce d, and sends $\langle SID_m, d\rangle$ to CS. Upon getting S_m's registration request, the control server CS computes $PSID_m = h\langle SID_m \parallel d\rangle$, $BS_m = h(PSID_m \parallel y)$, and sends $\langle BS_m\rangle$ to S_m over a secure channel. Finally, S_m keeps the secret data $\langle BS_m, d\rangle$ in a secure place (e.g., a usb-key).

4.1.2. User Registration Phase. When the user U_i registers, she first prefers an identity ID_i, password PW_i, and 2 random nonces $\langle b_1, b_2\rangle$. Then, she computes $A_i = h(PW_i \parallel b_1)$, $PID_i = h(ID_i \parallel b_2)$, $bb_i = b_2 \oplus A_i$, and sends $\langle A_i, PID_i\rangle$ to the CS securely. Once getting $\langle A_i, PID_i\rangle$, CS conducts the following steps:

$$\begin{aligned} C_i &= h\left(A_i \parallel PID_i\right) \\ D_i &= h\left(PID_i \parallel x\right) \\ E_i &= D_i \oplus A_i. \end{aligned} \tag{1}$$

Finally, CS stores the parameters $\{C_i, E_i, h(\cdot)\}$ in the smartcard and delivers the smartcard for U_i through a private communication channel. Then, U_i stores $\langle DP, bb_i\rangle$ in the card, where $DP = h(ID_i \parallel PW_i) \oplus b_1$. Finally, the smartcard stores $\langle C_i, E_i, bb_i, DP, h(\cdot)\rangle$.

4.1.3. Login Phase. U_i first inputs the smartcard into a card reader CR and keys ID_i^* and PW_i^* on the reader. Then, CR calculates

$$\begin{aligned} b_1^* &= DP \oplus h\left(ID_i^* \parallel PW_i^*\right) \\ A_i^* &= h\left(PW_i^* \parallel b_1^*\right) \\ b_2^* &= bb_i \oplus A_i^* \\ PID_i^* &= h\left(ID_i^* \parallel b_2^*\right) \\ C_i^* &= h\left(A_i^* \parallel PID_i^*\right). \end{aligned} \tag{2}$$

Then, the card reader CR checks whether $(C_i^* \stackrel{?}{=} C_i)$. If $(C_i^* == C_i)$, it means that $(ID_i^* = ID_i)$ and $(PW_i^* = PW_i)$. Then, CR produces a 128 bit random nonce N_i and computes

$$\begin{aligned} D_i &= E_i \oplus A_i \\ G_i &= h\left(PID_i \parallel SID_m \parallel N_i \parallel TS_i \parallel D_i\right) \\ F_i &= D_i \oplus N_i \\ Z_i &= SID_m \oplus h\left(D_i \parallel N_i\right) \end{aligned} \tag{3}$$

where SID_m is the identity of the cloud server from which by U_i wants to login. Then, CR sends the login request $\langle G_i, F_i, Z_i, PID_i, TS_i\rangle$ to S_m publicly.

4.1.4. Authentication Phase. This phase is to achieve mutual authentication and key agreement among U_i, S_m, and CS.

Step 1. S_m first verifies whether $(TS_m - TS_i < \triangle T)$, where TS_m is the cloud server's current timestamp and $\triangle T$ is the allowed transmission delay time. If the equality is not true, S_m rejects the connection; otherwise, S_m generates a 128 bit random nonce N_m and computes

$$\begin{aligned} J_i &= BS_m \oplus N_m \\ K_i &= h\left(N_m \parallel BS_m \parallel G_i \parallel TS_j\right). \end{aligned} \tag{4}$$

Then, S_m sends $\langle J_i, K_i, PSID_m, G_i, F_i, Z_i, PID_i, TS_i, TS_m\rangle$ to the CS over a public channel.

Step 2. On getting login messages, CS first checks whether $(TS_{cs} - TS_m < \triangle T^*)$, where TS_{cs} is the cloud server's current timestamp and $\triangle T^*$ the expected delay time for transmission. If the equality is not true, S_m rejects the connection; otherwise, CS computes

$$\begin{aligned} D_i &= h\left(PID_i \parallel x\right) \\ N_i^* &= F_i \oplus D_i \\ SID_m^* &= Z_i \oplus h\left(D_i \parallel N_i^*\right) \\ G_i^* &= h\left(PID_i \parallel SID_m^* \parallel N_i^* \parallel D_i \parallel TS_i\right). \end{aligned} \tag{5}$$

Then, CS checks whether $(G_i^* \stackrel{?}{=} G_i)$. If $(G_i^* == G_i)$, CS deems U_i as legal; otherwise, rejects the connection. Then, CS computes

$$\begin{aligned} BS_m^* &= h\left(PSID_m \parallel y\right) \\ N_m^* &= BS_m^* \oplus J_i \\ K_i^* &= h\left(BS_m^* \parallel N_m^* \parallel G_i \parallel TS_m\right). \end{aligned} \tag{6}$$

Once again, CS checks $(K_i^* \stackrel{?}{=} K_i)$. If $(K_i^* == K_i)$, CS deems S_m as legal; otherwise, it rejects the connection. Then, CS selects a 128 bit random nonce N_{cs} and computes

$$\begin{aligned} P_{cs} &= N_m \oplus N_{cs} \oplus h(N_i \parallel D_i) \\ R_{cs} &= N_i \oplus N_{cs} \oplus h(BS_m^* \parallel N_m^*) \\ SK_{cs} &= h(N_i \oplus N_m \oplus N_{cs}) \\ Q_{cs} &= h((N_m \oplus N_{cs}) \parallel SK_{cs}) \\ V_{cs} &= h((N_i \parallel N_{cs}) \parallel SK_{cs}) \end{aligned} \tag{7}$$

where SK_{cs} is the agreed session key. Then, CS sends $\langle P_{cs}, R_{cs}, Q_{cs}, V_{cs}\rangle$ to S_m.

Step 3. On getting the reply from CS, S_m calculates

$$\begin{aligned} W_m &= h(BS_m \parallel N_m) \\ N_i \oplus N_{cs} &= R_{cs} \oplus W_m \\ SK_m &= h(N_i \oplus N_{cs} \oplus N_m) \\ V_{cs}^* &= h((N_i \oplus N_{cs}) \parallel SK_m). \end{aligned} \tag{8}$$

Then, S_m verifies $V_{cs}^* \stackrel{?}{=} V_{cs}$ or not and if $V_{cs}^* \neq V_{cs}$, rejects the connection, and then sends $\langle P_{cs}, Q_{cs}\rangle$ to U_i publicly.

Step 4. On getting S_m, U_i calculates

$$\begin{aligned} L_i &= h(N_i \parallel D_i) \\ N_m \oplus N_{cs} &= P_{cs} \oplus L_i \\ SK_i &= h(N_m \oplus N_{cs} \oplus N_i) \\ Q_{cs}^* &= h((N_m \oplus N_{cs}) \parallel SK_i). \end{aligned} \tag{9}$$

Then, U_i verifies $(Q_{cs}^* \stackrel{?}{=} Q_{cs})$ and if $(Q_{cs}^* == Q_{cs})$, it demonstrates the authenticity of S_m and CS. Finally, mutual authentication are attained among U_i, S_m and CS. Now, U_i and S_m share the same session key $SK_m = SK_i$, and they can exchange sensitive information subsequently using $SK_m = SK_i$.

4.1.5. Password Change Phase. This phase is performed locally: not interacting with the server. When U_i wants to update her password, she provides ID_i and PW_i after inputting the smartcard. Then, CR executes the operations:

$$\begin{aligned} b_1^* &= DP \oplus h(ID_i^* \parallel PW_i^*) \\ A_i^* &= h(PW_i^* \parallel b_1^*) \\ b_2^* &= bb_i \oplus A_i^* \\ PID_i^* &= h(ID_i^* \parallel b_2^*) \\ C_i^* &= h(A_i^* \parallel PID_i^*). \end{aligned} \tag{10}$$

The smartcard verifies $(C_i^* \stackrel{?}{=} C_i)$. If $(C_i^* == C_i)$, user U_i is asked to enter a new password PW_i^{new} and calculates

$$\begin{aligned} A_i^{new} &= h(PW_i^{new} \parallel b_1) \\ C_i^{new} &= h(A_i^{new} \parallel PID_i^*) \\ D_i &= E_i \oplus A_i = h(PID_i^* \parallel x), \\ bb_i^* &= b_2^* \oplus A_i^{new} \\ E_i^{new} &= D_i \oplus A_i^{new} \\ DP^{new} &= h(ID_i \parallel PW_i^{new}) \oplus b_1^*. \end{aligned} \tag{11}$$

Finally, CR substitutes $\langle C_i, E_i, bb_i, DP^{new}\rangle$ with $\langle C_i^{new}, E_i^{new}, bb_i^*, DP^{new}\rangle$, respectively, in the card.

4.2. Cryptanalysis of Amin Et Al.'s Scheme. Recall that the three assumptions listed in Section 2.2.1 are also clearly made in Amin et al.'s scheme [21]. However, we observe that this scheme still remains feasible for an attacker to offline guess a user's password. This means that the primary goal of "truly two-factor security" cannot be satisfying. In addition, the scheme cannot provide forward secrecy and user untraceability.

4.2.1. Offline Password Guessing Attack. We find Amin et al.'s scheme cannot achieve truly two-factor security: it is subject to two types of offline password guessing attack when the smart card factor is compromised.

Type-I Attack. Suppose the secret parameters $\{C_i, E_i, bb_i, DP, h(\cdot)\}$ stored in U_i's smart card are somehow obtained by $\mathscr{A}$. At this point, $\mathscr{A}$ can find out U_i's identity and password as follows:

Step 1. Guesses U_i's identity ID_i^* and password PW_i^* from dictionary space $\mathscr{D}_{id}$ and $\mathscr{D}_{pw}$.

Step 2. Computes $b_1^* = DP \oplus h(ID_i^* \parallel PW_i^*)$, where DP is extracted from the smart card.

Step 3. Computes $A_i^* = h(PW_i^* \parallel b_1^*)$;

Step 4. Computes $b_2^* = bb_i \oplus A_i^*$, where bb_i is extracted from the smart card.

Step 5. Computes $PID_i^* = h(ID_i^* \parallel b_2^*)$;

Step 6. Computes $C_i^* = h(A_i^* \parallel PID_i^*)$;

Step 7. Checks the validity of (ID_i^*, PW_i^*) by comparing the computed C_i^* with C_i which is extracted from card.

Step 8. Repeats Steps 1~7 until determine the right pair of (ID_i^*, PW_i^*).

The time complexity of this attack is $\mathcal{O}(|\mathscr{D}_{id}| * |\mathscr{D}_{pw}| * 4T_H)$ [3, 16]. According to the running time in [16], $\mathscr{A}$ may complete the above attacking procedure within 35.2 days on a Laptop. This issue arises due to the inherent "usability-security tension": to achieve local password change (i.e., C-2 in [3]) and timely typo detection (i.e., C-9 in [3]); there is an

explicit password verifier $C_i = h(A_i \parallel PID_i) = h(h(PW_i \parallel b_1) \parallel h(ID_i \parallel b_2))$ stored in U_i's smart card, yet this verifier leads to a Type-I offline password attack.

To tackle this security issue while not losing usability, a viable countermeasure is to adopt the "fuzzy-verifier" technique [20] and keep $C_i = h(h(PW_i \parallel b_1) \parallel h(ID_i \parallel b_2) \bmod n)$ in U_i's smart card, where n confines the capacity of (ID, PW) pair, $2^4 \leq n \leq 2^8$. As a result, even if $\mathscr{A}$ obtains D_i^2, she can not identify the correct (ID, PW) from the above attack, because there will be $|\mathscr{D}_{id} * \mathscr{D}_{pw}|/n \approx 2^{32}$ candidate (ID, PW) pairs that make $D_i^{2*} = D_i^2$ in Step 4. To further identify the exactly correct (ID, PW) pair, $\mathscr{A}$ needs to interact with the server, and we can adopt the "honeywords" technique [3, 56] to confine $\mathscr{A}$'s advantage to a very limited value.

Type-II Attack. In the above attack, $\mathscr{A}$ does not need the protocol messages. In this attack, we assume that $\mathscr{A}$ can somehow obtain user's smart card and extract its secret parameters $\{C_i, E_i, bb_i, DP, h(\cdot)\}$ and can also eavesdrop the login messages $\{G_i, F_i, Z_i\}$ from the public channel. Now, $\mathscr{A}$ can *offline* guess U_i's password and identity simultaneously as follows:

Step 1. Guesses the value of ID_i^*, PW_i^* from dictionary space $\mathscr{D}_{id}$ and $\mathscr{D}_{pw}$.

Step 2. Computes $b_1^* = DP \oplus h(ID_i^* \parallel PW_i^*)$, where DP is extracted from the smart card.

Step 3. Computes $A_i^* = h(PW_i^* \parallel b_1^*)$;

Step 4. Computes $D_i^* = E_i \oplus A_i^*$.

Step 5. Computes $N_i^* = F_i \oplus D_i^*$;

Step 6. Computes $Z_i^* = SID_m \oplus h(D_i^* \oplus N_i^*)$;

Step 7. Checks the correctness of (ID_i^*, PW_i^*) by comparing if the computed Z_1^* equals the intercepted Z_1.

Step 8. Repeats Steps 1~7 of the above procedure until the correct value of (ID_i^*, PW_i^*) is found.

The time complexity of the above attacking procedure is $\mathcal{O}(|\mathscr{D}_{id}| * |\mathscr{D}_{pw}| * 3T_H)$, and $\mathscr{A}$ may complete the procedure within 26.4 days on a Laptop. In comparison, the Type-I attack is more practical. Amin et al. noted that user's mobile devices and IoT sensors are generally of "low memory, low power, and battery and network limitations" [21], and thus they prefer to only use lightweight symmetric key techniques to achieve robust security. It is just this motivation that makes their scheme vulnerable to the Type-II offline password guessing attack. And the rationales can be found in [20, 57].

4.2.2. No Forward Secrecy. A scheme that supports forward secrecy ensures that even after the long-term private key(s) of one or more participants was(were) leaked, previously agreed session keys remain secure [27]. This is important, especially when considering the serious situations of today's clouds like the compromise of cloud servers (e.g., [15]).

If an attacker $\mathscr{M}$ has obtained the cloud server S_m's long-term key BS_m from the breached server S_m and intercepted the messages $\{J_i; P_{cs}, R_{cs}, Q_{cs}, V_{cs}\}$ that are exchanged between U_i and S_m's authentication process from the public channel. $\mathscr{M}$ is able to figure out the session key using the following method:

Step 1. $\mathscr{M}$ computes $N_m = J_i \oplus BS_m$;

Step 2. $\mathscr{M}$ computes $W_m = h(BS_m \parallel N_m)$;

Step 3. $\mathscr{M}$ computes $N_i \oplus N_{cs} = R_{cs} \oplus W_m$;

Step 4. $\mathscr{M}$ calculates the session key $SK_m = h(N_i \oplus N_{cs} \oplus N_m)$.

With the session key SK_m computed, the entire session will be no secret to $\mathscr{M}$. This vulnerability is due to a result of violating the "forward secrecy principle" given in [20]: public-key technique is necessary to attain forward secrecy and at least two exponential operations (or point multiplications in ECC) are needed at the server side.

4.2.3. No User Untraceability. As revealed in [22], in the context of user authentication, there are two types of user anonymity: (1) the basic one, i.e., user identity-protection which requires that the attacker is unable to get the target user's real identity from eavesdropping the protocol messages; (2) the advanced one, i.e., user untraceability which guarantees that the attacker is unable to neither figure out the target user's identity nor determine whether two conversations are come from the same (unknown) user from eavesdropping the protocol messages. Therefore, most schemes (e.g., [4, 35, 39, 58–62]) that aim to preserve user anonymity attempt to fulfil the latter advanced notion of user anonymity. However, in Amin et al.'s scheme [21], the pseudoidentity PID_i is sent in every login request. Thus, the attacker can track all the login activities through this static pesudoidentity PID_i.

5. Revisiting Wei Et Al.'s Scheme

In 2018, Wei et al. [23] observed that most of previous schemes (e.g., [36–40]) only provided some heuristic security arguments and little attention has been to the formal treatment of protocol security. Accordingly, they employed an authenticated encryption scheme to construct a new two-factor scheme that can provide provable security in the random oracle model. Though this scheme is armed with a formal proof, we will show that this scheme still has several serious defects being overlooked: (1) it cannot achieve the primary goal of "truly two-factor security"; (2) it provides no forward secrecy; (3) it ensures no user untraceability.

5.1. Review Wei Et Al.'s Scheme. There are three phases in et al.'s scheme: registration, authentication and key exchange, and password update.

5.1.1. Registration Phase. When the user U_i registers, she first prefers an identity ID_i, password PW_i, and a random nonces b; then U_i calculates $PW_i^* = h(PW_i \parallel b)$ and sends $\{ID_i, PW_i^*\}$ to the gateway node GW via a secure channel. On receiving $\{ID_i, PW_i^*\}$, GW computes $V_i = h(ID_i \parallel K)$ and $N_i = V_i \oplus h(ID_i \parallel PW_i^*)$. GW also selects U_i's pseudoidentity DID_i and keeps the item (DID_i, ID_i) in its database. Finally, GW produces a smart card with security parameters $(DID_i, N_i, h(\cdot))$ and sends to U_i the card via a trusted channel. Then, U_i adds the random number b into the card.

5.1.2. Authentication and Key Exchange Phase. U_i aims to access the service from the cloud server S_j, U_i executes the authentication and key exchange phase with GW and S_j as follows.

(1) U_i inserts her card into the card reader, keys her identity ID_i, and password PW_i. The card computes $PW_i^* = h(PW_i \parallel b)$ and gets $V_i^* = N_i \oplus h(ID_i \parallel PW_i^*)$. Then, the card calculates a key $k_1 = h(V_i^* \parallel T_1)$, where T_1 is the current timestamp in U_i's system. The card also selects a random nonce $R_1 \in \{0,1\}^l$ and encrypts R_1 using an authenticated encryption scheme to obtain $C_1 = E_{k_1}^{T_1, DID_i}(R_1)$. Finally, the card transmits the message (DID_i, S_j, C_1, T_1) to GW.

(2) Upon receiving (DID_i, S_j, C_1, T_1) at time T_1^*, GW verifies whether $T_1^* - T_1 \leq \Delta T$ or not, where ΔT stands for the largest time interval allowed for the transmission delay. If the check passes, GW retrieves U_i's real identity ID_i in the database using DID_i and calculates $k_1 = h(h(ID_i \parallel K) \parallel T_1)$. GW then decrypts C_1 and recovers $R_1 = D_{k_1}^{T_1, DID_i}(C_1)$. If the decryption fails, the connection is rejected; otherwise, U_i is verified by GW. GW calculates a key $k_2 = h(ID \parallel h(K \parallel S_j) \parallel T_2)$, where T_2 is the current timestamp at GW. Then, GW encrypts R_1 using an authenticated encryption scheme to obtain $C_2 = E_{k_2}^{T_2, ID}\{R_1\}$. Finally, GW transmits (ID, C_2, T_2) to the server S_j.

(3) On getting (ID, C_2, T_2) at time T_2^*, the server S_j verifies if $T_2^* - T_2 \leq \Delta T$. If it is true, S_j calculates $k_2 = h(ID \parallel x_j \parallel T_2)$ and decrypts C_2 and calculates $R_1 = D_{k_2}^{T_2, ID}\{C_2\}$. If the decryption fails, the connection is rejected; otherwise, S_j computes $k_3 = h(ID \parallel R_1 \parallel T_3)$, where T_3 is the current timestamp at S_j. Then, S_j selects a random number $R_2 \in \{0,1\}^l$ and encrypts R_2 using k_3 to get $C_3 = E_{k_3}^{T_3, ID}\{R_2\}$. S_j transmits (C_3, T_3) to U_i. Finally, S_j calculates the session key $sk = h(ID \parallel T_3 \parallel R_1 \parallel R_2)$ and accepts U_i.

(4) On getting (C_3, T_3) at time T_3^*, U_i checks if $T_3^* - T_3 \leq \Delta T$. If it holds, GW calculates $k_3 = h(ID \parallel R_1 \parallel T_3)$ and decrypts $R_2 = D_{k_3}^{T_3, ID}\{C_4\}$. If the decryption does not fail, U_i deems the session as valid and computes $sk = h(ID \parallel T_3 \parallel R_1 \parallel R_2)$.

Finally, U_i and S_j share the common session key sk.

5.1.3. Password Updating Phase. This phase is executed when U_i aims to update her password PW_i with a new one. Because this phase has little relevance with the following cryptanalysis, we omit it.

5.2. Cryptanalysis of Amin Et Al.'s Scheme. Recall that the three assumptions listed in Section 2.2.1 are also clearly made in Wei et al.'s scheme [23]. Though this scheme enjoys many desirable attributes such as rigorous security model and the introduction of an authenticated encryption scheme (i.e., Hoang et al.'s AEZ [63]), we now show its defects.

5.2.1. Offline Password Guessing Attack. We find Amin *et al.*'s scheme cannot achieve truly two-factor security: it is subject to Type-II offline password guessing attack as illustrated in Section 4.2.1.

Type-II Attack. In this attack, we assume that $\mathcal{A}$ can somehow obtain user's smart card and extract its secret parameters $\{DID_i, N_i, b, h(\cdot)\}$ and also can eavesdrop the login messages $\{DID_i, S_j, C_1, T_1\}$ from the public channel. Now, $\mathcal{A}$ can *offline* guess U_i's password and identity simultaneously as follows:

Step 1. Guesses the value of ID_i', PW_i' from dictionary space $\mathcal{D}_{id}$ and $\mathcal{D}_{pw}$.

Step 2. Computes $PW_i^{*\prime} = h(PW_i' \parallel b)$, where b is extracted from the smart card.

Step 3. Computes $V_i' = N_i \oplus h(ID_i' \parallel PW_i^{*\prime})$, where N_i is extracted from the smart card.

Step 4. Computes $k_1' = h(V_i' \parallel T_1)$;

Step 5. Decrypts C_1 with k_1';

Step 6. If the above decryption succeeds, it indicates that ID_i' and PW_i' are correct and terminate.

Step 7. Repeats Steps 1~5 of the above procedure until find the correct value of (ID_i, PW_i).

The time complexity of the above attacking procedure is $\mathcal{O}(|\mathcal{D}_{id}| * |\mathcal{D}_{pw}| * 3T_H)$, and $\mathcal{A}$ may finish the attack within 26.4 days on a common PC. In comparison, the Type-I attack is more realistic. Wei et al. noted that user's mobile devices and IoT sensors are generally resource-constrained and made an attempt to design a secure and efficient scheme "without using computation-expensive public key operations" [23]. It is just this motivation that makes their scheme vulnerable to the Type-II offline password guessing attack. And the rationales are referred to [20, 57].

5.2.2. No Forward Secrecy. As with Amin et al.'s scheme [21], Wei et al.'s scheme also only employs symmetric key techniques and thus violates the "forward secrecy principle" proposed in [20], and see more in Section 4.2.2. We now show the details.

Table 2: Summary of our cryptanalysis results and the underlying vulnerabilities.

Scheme	Weaknesses	Vulnerabilities
Wu-Xu [17]	Offline password guessing attack	When using Elgamal-like encryption (e.g., chaotic-based), at least two exponentiations are needed at the user side [18]
	Replay attack	Their scheme employs random numbers but has only two protocol flows
Roy et al. [19]	Offline password guessing attack	No public-key technique used [20]
	No forward secrecy	No public-key technique used [20]
	Poor scalability	Poor design
Amin et al. [21]	Offline password guessing attack	No public-key technique used [20]
	No forward secrecy	No public-key technique used [20]
	No user un-traceability	No public-key technique used [22]
Wei et al. [23]	Offline password guessing attack	No public-key technique used [20]
	No forward secrecy	No public-key technique used [20]
	No user un-traceability	No public-key technique used [22]

If an attacker $\mathcal{M}$ has obtained the cloud server S_j's long-term key x_j from the breached server S_j and intercepted the messages $\{ID, C_2, T_2, T_3, C_3\}$ that are exchanged among U_i, GW, and S_j's authentication process from the public channel, $\mathcal{M}$ is able to figure out the session key using the following method:

Step 1. $\mathcal{M}$ computes $k_2 = h(ID \parallel x_j \parallel T_2)$;

Step 2. $\mathcal{M}$ decrypts C_2 using k_2 to obtain $R_1 = D_{k_2}^{T_2, ID}\{C_2\}$;

Step 3. $\mathcal{M}$ calculates $k_3 = h(ID \parallel R_1 \parallel T_3)$;

Step 4. $\mathcal{M}$ decrypts C_3 using k_3 to obtain $R_2 = D_{k_3}^{T_3, ID}\{C_3\}$;

Step 5. $\mathcal{M}$ calculates the session key $sk = h(ID \parallel T_3 \parallel R_1 \parallel R_2)$.

5.2.3. No User Un-Traceability. As with Amin et al.'s scheme [21], Wei et al.'s scheme also only employs the pseudo-identity technique to preserve user anonymity, and this breaches the "anonymity principle" proposed in [22]: under the non-tamper resistant assumption of smart cards, public-key primitives are necessary for achieving user un-traceability for two-factor user authentication schemes. In Wei *et al.*'s scheme [21], the static pseudo-identity DID_i is sent in every login request. Thus, the attacker can track all the login activities through this static pesudo-identity DID_i.

6. Conclusion

A large amount of efforts have been devoted to the design of smart-card-based password authentication scheme, yet it still remains an open challenge to devise an efficient, secure, and privacy-preserving scheme under the assumption that smart cards can be tampered. Very recently, Wu-Xu, Roy et al., Amin et al., and Wei et al. made four other attempts. However, through careful analysis we show that all of them still have several serious drawbacks being overlooked (see Table 2), and most these drawbacks are due to violation of basic protocols design principles (e.g., [3, 20]). Taking our attacks in mind, we are considering designing an efficient two-factor scheme with provable security.

Data Availability

Data sharing is not applicable to this article as no new data was created or analyzed in this study.

Disclosure

Part of this paper was presented at the 4th International Conference on Cloud Computing and Security (ICCCS 2018) [14].

Conflicts of Interest

The authors declare that no conflicts of interest exist.

Acknowledgments

This research was in part supported by the National Key Research and Development Plan under Grant No. 2016YFB0800600, by the National Natural Science Foundation of China under Grant No. 61802006, by the National Key Research and Development Plan under Grant No. 2017YFB1200700, and by China Postdoctoral Science Foundation under Grant No. 2018M640026.

References

[1] Z. Hao, S. Zhong, and N. Yu, "A time-bound ticket-based mutual authentication scheme for cloud computing," *International Journal of Computers, Communications and Control*, vol. 6, no. 2, pp. 227–235, 2011.

[2] J. Yu, G. Wang, Y. Mu, and W. Gao, "An efficient generic framework for three-factor authentication with provably secure

instantiation," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2302–2313, 2014.

[3] D. Wang and P. Wang, "Two birds with one stone: two-factor authentication with security beyond conventional bound," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 708–722, 2018.

[4] Q. Xie, D. S. Wong, G. Wang, X. Tan, K. Chen, and L. Fang, "Provably secure dynamic ID-based anonymous two-factor authenticated key exchange protocol with extended security model," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1382–1392, 2017.

[5] V. Odelu, A. K. Das, S. Kumari, X. Huang, and M. Wazid, "Provably secure authenticated key agreement scheme for distributed mobile cloud computing services," *Future Generation Computer Systems*, vol. 68, pp. 74–88, 2017.

[6] R. Amin, S. H. Islam, P. Gope, K. R. Choo, and N. Tapas, "Anonymity preserving and lightweight multi-medical server authentication protocol for telecare medical information system," *IEEE Journal of Biomedical and Health Informatics*, p. 1, 2018.

[7] Q. Jiang, Y. Qian, J. Ma, X. Ma, Q. Cheng, and F. Wei, "User centric three-factor authentication protocol for cloud-assisted wearable devices," *International Journal of Communication Systems*, vol. 32, pp. 1–20, 2019.

[8] Q. Jiang, M. K. Khan, X. Lu, J. Ma, and D. He, "A privacy preserving three-factor authentication protocol for e-Health clouds," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3826–3849, 2016.

[9] A. K. Das, M. Wazid, N. Kumar, M. K. Khan, K. R. Choo, and Y. Park, "Design of secure and lightweight authentication protocol for wearable devices environment," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 4, pp. 1310–1322, 2018.

[10] D. Wang, Q. Gu, H. Cheng, and P. Wang, "The request for better measurement: A comparative evaluation of two-factor authentication schemes," in *Proceedings of the ACM, ASIACCS '16*, pp. 475–486, 2016.

[11] H. Xiong, J. Tao, and C. Yuan, "Enabling telecare medical information systems with strong authentication and anonymity," *IEEE Access*, vol. 5, pp. 5648–5661, 2017.

[12] D. Wang and P. Wang, "On the usability of two-factor authentication," in *Proceedings of the SecureComm*, pp. 141–150, 2014.

[13] G. Xu, J. Liu, Y. Lu, X. Zeng, Y. Zhang, and X. Li, "A novel efficient MAKA protocol with desynchronization for anonymous roaming service in Global Mobility Networks," *Journal of Network and Computer Applications*, vol. 107, pp. 83–92, 2018.

[14] Y. Shen, D. Wang, and P. Wang, "Revisiting anonymous twofactor authentication schemes for cloud computing," in *Proceedings of the ICCCS*, pp. 134–146, 2018.

[15] All Data Breach Sources, https://breachalarm.com/all-sources, 2019.

[16] D. Wang, D. He, P. Wang, and C.-H. Chu, "Anonymous twofactor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 4, pp. 428–442, 2015.

[17] F. Wu and L. Xu, "A chaotic map-based authentication and key agreement scheme with user anonymity for cloud computing," in *Proceedings of the ICCCS '17*, pp. 189–200, Springer, 2017.

[18] D. Wang, N. Wang, P. Wang, and S. Qing, "Preserving privacy for free: efficient and provably secure twofactor authentication scheme with user anonymity," *Information Sciences*, vol. 321, pp. 162–178, 2015.

[19] S. Roy, S. Chatterjee, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "On the design of provably secure lightweight remote user authentication scheme for mobile cloud computing services," *IEEE Access*, vol. 5, no. 25, pp. 808–25825, 2017.

[20] C.-G. Ma, D. Wang, and S.-D. Zhao, "Security flaws in two improved remote user authentication schemes using smart cards," *International Journal of Communication Systems*, vol. 27, no. 10, pp. 2215–2227, 2014.

[21] R. Amin, N. Kumar, G. P. Biswas, R. Iqbal, and V. Chang, "A light weight authentication protocol for IoT-enabled devices in distributed Cloud Computing environment," *Future Generation Computer Systems*, vol. 78, pp. 1005–1019, 2018.

[22] D. Wang and P. Wang, "On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions," *Computer Networks*, vol. 73, pp. 41–57, 2014.

[23] F. Wei, R. Zhang, and C. Ma, "A provably secure anonymous two-factor authenticated key exchange protocol for cloud computing," *Fundamenta Informaticae*, vol. 157, no. 1, pp. 201–220, 2018.

[24] G. Yang, D. S. Wong, H. Wang, and X. Deng, "Two-factor mutual authentication based on smart cards and passwords," *Journal of Computer and System Sciences*, vol. 74, no. 7, pp. 1160–1172, 2008.

[25] 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 Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1767–1775, 2014.

[26] X. Li, J. Niu, J. Liao, and W. Liang, "Cryptanalysis of a dynamic identity-based remote user authentication scheme with verifiable password update," *International Journal of Communication Systems*, vol. 28, no. 2, pp. 374–382, 2015.

[27] C. Wang and G. Xu, "Cryptanalysis of three passwordbased remote user authentication schemes with nontamper resistant smart card," *Security and Communication Networks*, vol. 2017, Article ID 1619741, 14 pages, 2017.

[28] J. Xu, W.-T. Zhu, and D.-G. Feng, "An improved smart card based password authentication scheme with provable security," *Computer Standards & Interfaces*, vol. 31, no. 4, pp. 723–728, 2009.

[29] S. K. Sood, A. K. Sarje, and K. Singh, "An improvement of xu et al.'s authentication scheme using smart cards," in *Proceedings of the ACM COMPUTE '10*, pp. 1–5, ACM, 2010.

[30] R. Song, "Advanced smart card based password authentication protocol," *Computer Standards & Interfaces*, vol. 32, no. 5, pp. 321–325, 2010.

[31] B.-L. Chen, W.-C. Kuo, and L.-C. Wuu, "Robust smart-card-based remote user password authentication scheme," *International Journal of Communication Systems*, vol. 27, no. 2, pp. 377–389, 2014.

[32] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proceedings of the CRYPTO 1999*, vol. 1666 of *LNCS*, pp. 388–397, Springer, 1999.

[33] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 541–552, 2002.

[34] J.-L. Tsai and N.-W. Lo, "A privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE Systems Journal*, vol. 9, no. 3, pp. 805–815, 2015.

[35] K.-P. Xue, P.-L. Hong, and C.-S. Ma, "A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture," *Journal of Computer and System Sciences*, vol. 80, no. 1, pp. 195–206, 2014.

[36] X. Li, Y. Xiong, J. Ma, and W. Wang, "An enhanced and security dynamic identity based authentication protocol for multi-server architecture using smart cards," *Journal of Network and Computer Applications*, vol. 35, no. 2, pp. 763–769, 2012.

[37] B. Wang and M. Ma, "A smart card based efficient and secured multi-server authentication scheme," *Wireless Personal Communications*, vol. 68, no. 2, pp. 361–378, 2013.

[38] D. He, Y. Gao, S. Chan, C. Chen, and J. Bu, "An enhanced two-factor user authentication scheme in wireless sensor networks," *Ad Hoc & Sensor Wireless Networks*, vol. 10, no. 4, pp. 361–371, 2010.

[39] M. L. Das, "Two-factor user authentication in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1086–1090, 2009.

[40] D.-Z. Sun, J.-X. Li, Z.-Y. Feng, Z.-F. Cao, and G.-Q. Xu, "ON the security and improvement of a two-factor user authentication scheme in wireless sensor networks," *Personal and Ubiquitous Computing*, vol. 17, no. 5, pp. 895–905, 2013.

[41] D. Dolev and A. C.-C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[42] Q. Jiang, J. Ma, G. Li, and X. Li, "Improvement of robust smart-card-based password authentication scheme," *International Journal of Communication Systems*, vol. 28, no. 2, pp. 383–393, 2014.

[43] X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiah, and S. Kumari, "A robust ECC based provable secure authentication protocol with privacy protection for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3599–3609, 2018.

[44] X. Li, J. Niu, S. Kumari, F. Wu, A. K. Sangaiah, and K. R. Choo, "A three-factor anonymous authentication scheme for wireless sensor networks in internet of things environments," *Journal of Network and Computer Applications*, vol. 103, pp. 194–204, 2018.

[45] D. Wang, P. Wang, H. Debiao, and Y. Tian, "A security analysis of honeywords," in *Proceedings of the Usenix Security*, pp. 1–18, 2019.

[46] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in *Proceedings of the IEEE S&P*, pp. 538–552, 2012.

[47] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, "Targeted online password guessing: An underestimated threat," in *Proceedings of the ACM CCS*, pp. 1242–1254, 2016.

[48] D. Wang and P. Wang, "Offline dictionary attack on password authentication schemes using smart cards," in *Proceedings of the ISC*, pp. 221–237, 2013.

[49] S. H. Islam, "Design and analysis of an improved smartcard-based remote user password authentication scheme," *International Journal of Communication Systems*, vol. 29, no. 11, pp. 1708–1719, 2016.

[50] M. Alsaleh, M. Mannan, and P. C. van Oorschot, "Revisiting defenses against large-scale online password guessing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 128–141, 2012.

[51] P. A. Grassi, E. M. Newton, R. A. Perlner et al., "NIST 800-63B digital identity guidelines: Authentication and lifecycle management," Tech. Rep., McLean, Va, USA, 2017.

[52] AD. Jaggard and P. Syverson, "Oft target," in *Proceedings of the PET*, pp. 1-2, 2017.

[53] B. Lu, X. Zhang, Z. Ling, Y. Zhang, and Z. Lin, "A measurement study of authentication rate-limiting mechanisms of modern websites," in *Proceedings of the ACSAC*, pp. 89–100, 2018.

[54] H. Debiao, C. Jianhua, and H. Jin, "An ID-based client authentication with key agreement protocol for mobile client-server environment on ECC with provable security," *Information Fusion*, vol. 13, no. 3, pp. 223–230, 2012.

[55] N. Memon, "How biometric authentication poses new challenges to our security and privacy [in the spotlight]," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 194–196, 2017.

[56] D. Wang, H. Cheng, P. Wang, J. Yan, and X. Huang, "A security analysis of honeywords," in *Proceedings of the NDSS*, ISOC, pp. 1–16, 2018.

[57] D. Wang and P. Wang, "Understanding security failures of two-factor authentication schemes for real-time applications in hierarchical wireless sensor networks," *Ad Hoc Networks*, vol. 20, pp. 1–15, 2014.

[58] X. Li, W. Qiu, D. Zheng, K. Chen, and J. Li, "Anonymity enhancement on robust and efficient passwordauthenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, pp. 793–800, 2010.

[59] P. Gope, A. K. Das, N. Kumar, and Y. Cheng, "Lightweight and physically secure anonymous mutual authentication protocol for real-time data access in industrial wireless sensor networks," *IEEE Transactions on Industrial Informatics*, 2019.

[60] A. Gupta, M. Tripathi, T. J. Shaikh, and A. Sharma, "A lightweight anonymous user authentication and key establishment scheme for wearable devices," *Computer Networks*, vol. 149, pp. 29–42, 2019.

[61] A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. Rodrigues, "Biometrics-Based Privacy-Preserving User Authentication Scheme for Cloud-Based Industrial Internet of Things Deployment," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4900–4913, 2018.

[62] T.-T. Truong, M.-T. Tran, and A.-D. Duong, "Improved chebyshev polynomials-based authentication scheme in client-server environment," *Security and Communication Networks*, vol. 2019, Article ID 4250743, 11 pages, 2019.

[63] V. T. Hoang, T. Krovetz, and P. Rogaway, "Robust authenticated-encryption aez and the problem that it solves," in *Proceedings of the EUROCRYPT*, pp. 15–44, 2015.