SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

January 2022, Vol. 65 112301:1–112301:15
https://doi.org/10.1007/s11432-020-2975-6

Efficient privacy-preserving user authentication scheme with forward secrecy for industry 4.0

Chenyu WANG[1,2], Ding WANG[3,4,2*], Guoai XU[1] & Debiao HE[5]

[1]*School of Cyber Security, Beijing University of Posts and Telecommunications, Beijing* 100876*, China;*
[2]*State Key Laboratory of Cryptology, P.O. Box* 5159*, Beijing* 100878*, China;*
[3]*College of Cyber Science, Nankai University, Tianjin* 300350*, China;*
[4]*Tianjin Key Laboratory of Network and Data Security Technology, Nankai University, Tianjin* 300350*, China;*
[5]*School of Cyber Science and Engineering, Wuhan University, Wuhan* 430072*, China*

Received 26 February 2020/Revised 11 May 2020/Accepted 2 July 2020/Published online 20 August 2021

Abstract Industry 4.0, which combines information technology, network and industrial production, is expected to have a tremendous impact on our daily lives. In such a complex and security-critical system with resource-constrained sensor nodes, the design of a secure user authentication scheme for preventing real-time data from unauthorized access is full of challenges, and the main crux lies in how to realize the important property of forward secrecy. Existing schemes either fail to achieve forward secrecy or achieve forward secrecy with high computation cost on sensor nodes. Besides, they often fail to conform to the development trend of industry 4.0 systems where a cloud center is necessary to help intelligent decision-making and alleviate computation and storage pressure. Therefore, in this paper, we propose an efficient privacy-preserving user authentication scheme with forward secrecy for industry 4.0, and formally prove its security in the random oracle model. Compared with previous schemes, it has three advantages: (1) all eleven state-of-the-art criteria are achieved; (2) its computation cost on sensor nodes is comparable to those insecure schemes that employ only symmetric cryptographic algorithms, and is superior to those that also use asymmetric cryptographic algorithms; (3) it takes the advantage of the computation and storage capabilities of the cloud center to achieve user anonymity and the resistance to offline dictionary attack without performing any asymmetric cryptographic algorithms on gateways. Our computation cost on gateways is the smallest among all state-of-the-art relevant schemes for comparison.

Keywords industry 4.0, wireless sensor networks, password authentication, forward secrecy, offline dictionary attack

Citation Wang C Y, Wang D, Xu G A, et al. Efficient privacy-preserving user authentication scheme with forward secrecy for industry 4.0. Sci China Inf Sci, 2022, 65(1): 112301, https://doi.org/10.1007/s11432-020-2975-6

1 Introduction

Industry 4.0 is "fully-integrated, collaborative manufacturing system that responds in real-time to meet changing demands and conditions in the factory, in the supply network and in customer needs" [1]. It is a new phase of industrial revolution, and offers a more comprehensive, interlinked and holistic approach for manufacturing by taking full advantage of the combination of information communication technology and information-physical systems [2]. As shown in Figure 1, one of the critical network infrastructures of industry 4.0 is the wireless sensor network (WSN). It is used in industry 4.0 systems with cloud computing technology [3–5] to collect data from environment and automatically build learning models for promoting industrial intelligence. Generally, a WSN is formed by lots of distributed sensor nodes and a gateway node. The sensor nodes are resource-constrained devices with limited computation and storage capabilities [6–8].

In some industry 4.0 applications, users may need to access the real-time data stored in sensor nodes. For example, in a production maintenance system of industry 4.0, normally, the sensor nodes will monitor the key parts of industrial machines and submit the collected data to the cloud center periodically. Then,

* Corresponding author (email: wangding@nankai.edu.cn)

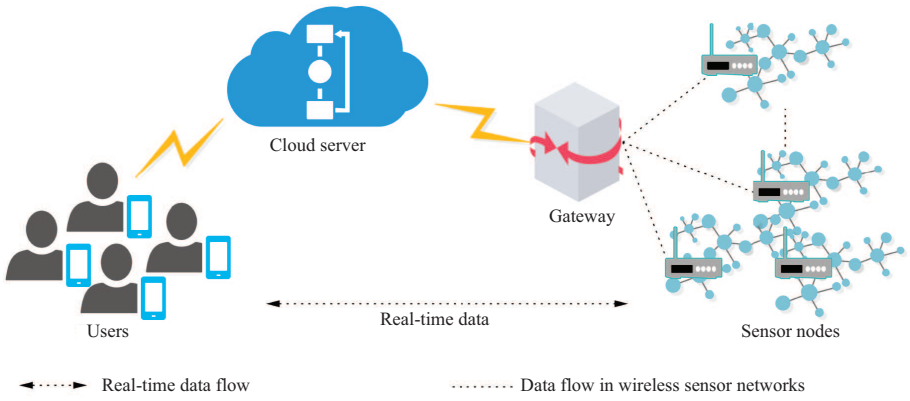

Figure 1 (Color online) System architecture of industry 4.0.

the data are analyzed, the use status of the machines is to be judged, and whether it needs to be sent to professional staff for maintenance is determined. Once there is any problem with the machines, the cloud center can make decisions timely to avoid an emergency shutdown. In this application, it is natural that the user (professional staff) wants to directly access the real-time data in a specific sensor node to further diagnose the system's failures or check the system's maintenance status by sampling. Generally, these industry 4.0 applications are security-critical with massive amounts of sensitive data, and thus they are the target of many adversaries. As such, it is important to protect sensitive data stored in sensor nodes from unauthorized access, an authentication mechanism is required to identify the authenticity of users, and a session key is necessary to be provided to secure their conversations. In a user authentication mechanism, there are three factors to authenticate a user: (1) something that the user knows, such as a password [9]; (2) something that the user has, such as a smart card [10,11]; (3) something that the user is, such as biometrics or behavior trait [12,13]. A multi-factor user authentication scheme combines at least two factors.

One of the important attributes of user authentication for industry 4.0 is forward secrecy (also known as perfect forward secrecy). It guarantees that the agreed session keys are not compromised even if the private key of gateways or sensor nodes is compromised. It can protect the past conversations against future compromises of secret keys [14]. Prior to the implementation of forward secrecy, all data transmitted between a user and a sensor node could be compromised if the private keys of gateways or sensor nodes were ever disclosed. One famous example of such an attack is the Heartbleed bug[1)], which allows the adversary to read the memory of the systems. Forward secrecy greatly improves systems' ability to resist risks. It ensures that, in the worst case, at least previous conversations are secure. New security standards, such as WiFi WPA3[2)] and TLS 1.3[3)], begin to take forward secrecy as an important feature of authentication protocols.

1.1 Related work

In 2009, the first smart-card-based password authentication protocol for WSNs was proposed by Das [15], where no public-key algorithm is involved. However, their scheme was shortly shown that it is subject to offline dictionary attack and fail to build a session key to protect conversations between users and sensor nodes. Then Fan et al. [16] designed a new privacy-preserving scheme using hash and exclusive-OR operations. Unfortunately, Wang et al. [17] proved that this scheme cannot preserve forward secrecy and suffers from an various attacks. Because their sensor nodes have the same secret key, and the session key is an outcome of the secret key and parameters transmitted in an open channel, the adversary who gets the secret key can get the session key and break the forward secrecy successfully. Next, Das et al. [18] presented a new user authentication scheme that supports dynamic node addition. Once again, this scheme subsequently was found to have various security problems, including failing to achieve forward secrecy. For example, in their scheme, all parameters that are contained in the session key are encrypted

1) The Heartbleed bug. http://heartbleed.com/.

2) Rescorla E. The transport layer security (TLS) protocol version 1.3, 2018. https://datatracker.ietf.org/doc/rfc8446/?include_text=1.

3) Kastrnakes J. Wi-Fi security is starting to get its biggest upgrade in over a dacade. 2018. https://www.theverge.com/circuitbreaker/2018/6/26/17501594/wpa3-wifi-security-certification.

using the secret key of the target sensor node. Therefore, the adversary who obtains the key can decrypt to get these parameters and then compute the session key correctly. In 2013, Xue et al. [19] proposed a secure scheme for WSNs, but failed again. To provide forward secrecy, they let users and sensor nodes choose a random number and jointly produce the session keys. However, the adversary who obtains the gateway's secret key can successfully calculate the session key in combination with the parameters in the public channel, thereby invalidating forward secrecy. Furthermore, Wang et al. [17] show that their scheme cannot preserve user anonymity. In conclusion, researchers are committed to breaking previous schemes and then proposing new secure protocols using lightweight symmetric cryptographic techniques. However, most attempts have failed and most schemes cannot achieve forward secrecy.

Recently, Wang et al. [17] and Ma et al. [20] demonstrate that lightweight public-key algorithms are indispensable to resist offline dictionary attacks and realize user anonymity in multi-factor user authentication protocols. Besides, to provide forward secrecy, the sensor nodes have to conduct at least two modular exponentiation or point multiplication operations. Following these principles, many public-key-algorithm-based schemes are proposed. For example, in 2014, Choi et al. [21] presented an authentication scheme based on elliptic-curve-cryptography (ECC), but their scheme cannot achieve user anonymity and suffers from an offline dictionary attacks. In 2015, He et al. [22] designed a discrete-logarithm-based authentication scheme, but it cannot preserve user anonymity and forward secrecy. In 2016, Reddy et al. [23] also presented an ECC-based three-factor authentication scheme. Unfortunately, it still cannot resist offline dictionary attacks, and its computational cost on sensor nodes and gateways are still high. In 2017, Jiang et al. [24] designed an improved scheme that aims to overcome offline dictionary attacks, but their attempt failed again. In 2018, an ECC-based provable secure scheme was introduced by Li et al. [3], but it is not secure against an offline dictionary attack and achieves forward secrecy with two point multiplication operations at sensor nodes. In brief, some of the schemes which employ public-key algorithms to enhance the security, still have this or that weaknesses, and they always have high computational cost: the sensor nodes need to run two modular exponentiation or point multiplication operations, and the gateway needs to run one modular exponentiation or point multiplication operation.

1.2 Motivations and contributions

Research over the past decade has shown that designing a secure user authentication protocol for an industry 4.0 environment is very difficult. Generally, sensor node has limited resources and it is difficult to perform computational expensive cryptographic primitives. On the other hand, the application scenarios of industry 4.0 often have high security and performance requirements on user authentication schemes. In particular, among many security requirements, the resource-constrained nature of sensor nodes brings greater challenges to the realization of forward secrecy. In addition, as the number of remote users and connected sensor nodes increases, the number of concurrent sessions becomes larger, and then the lack of computation and storage capabilities of gateways become more apparent. However, we find that most of the existing protocols only use lightweight symmetric cryptographic algorithms to cater to the resources-constrained nature of sensor devices, but this type of protocols violate the protocol design principles (i.e., asymmetric cryptographic algorithms are indispensable to design secure protocols [20, 25]), so these protocols are inherently unable to achieve forward secrecy and resist offline dictionary attacks. Remaining protocols that follow the protocol design principles employ asymmetric cryptographic algorithms to achieve better security, but such protocols need to perform at least two modular exponentiation or elliptic curve point multiplication operations at sensor nodes, and at least one modular exponentiation or elliptic curve point multiplication operation at gateways, which are inefficient.

Besides, the cloud center is usually the key computing center for intelligent decision-making of the entire system in industry 4.0 systems. Users often need the cloud center to provide various services to assist industrial intelligent production. Therefore, user authentication protocols can leverage the powerful computation and storage capabilities of the cloud center to well balance the security and performance. However, existing related protocols do not take the cloud center into account, thus they do not conform to the development trend of industry 4.0 systems in terms of system architecture, nor do they take the advantage of the powerful capabilities of the cloud center to alleviate the computing pressure of gateways.

Therefore, this paper attempts to propose a user authentication scheme that can cater to the development trend of industry 4.0 systems, balancing the conflict between the security and performance, so that users are able to access real-time data stored in sensor nodes securely. Our contributions are threefold.

(1) We depict the multi-factor user authentication model for industry 4.0 systems, this authentication

Table 1 Notations and abbreviations

Symbol	Description	Symbol	Description
U_i	The ith user	S_j	The jth sensor node
GWN_k	The kth gateway	GID_k	The identity of GWN_k
$\mathcal{A}$	The adversary	CS	The cloud center
ID_i	Identity of U_i	$\mathrm{Enc/Dec}(\cdot)$	Encryption/decryption function
PW_i	Password of U_i	$\mathrm{Gen/Rep}(\cdot)$	Fuzzy extractor
fng_i	Biometric of U_i	$h(\cdot)$	One-way hash function
SID_j	Identity of SN_j	x	CS's long term secret key
X_{S_j}	Secret key of S_j	x_{G_k}	GWN_k's long term secret key
X_{G_k}	Secret key of GWN_k	SK	Session key
$\oplus$	XOR operation	$\|$	Concatenation operation
$\rightarrow$	A public channel	$\Longrightarrow$	A secure channel

model can take the full advantage of the computation and storage capabilities of the cloud center and it is suitable for industry 4.0 environments.

(2) We exploit the imbalanced computational nature of the Rivest Shamir Adleman (RSA) cryptosystem to provide forward secrecy without expensive computation cost on the sensor nodes. Through taking advantages of the computation and storage capabilities of the cloud center, we achieve user anonymity and the resistance to offline dictionary attack without performing asymmetric cryptographic algorithms on the gateway.

(3) We formally prove the scheme's security under the random oracle model and compare it with relevant typical authentication schemes. The results demonstrate that our scheme meets all eleven state-of-the-art evaluation criteria with minimal computation cost on the gateway and very small computation cost on the sensor nodes, is especially suitable for industry 4.0 applications with a large number of resource-constrained sensor nodes.

2 Preliminaries

This section shows the authentication architecture, adversary model, and evaluation criteria of our proposed scheme. Note that Table 1 summarizes the notations and abbreviations of this paper.

2.1 Authentication architecture

As shown in Figure 2, to ensure that real-time data can be accessed securely by users, four participants are involved in user authentication schemes: the users with a smart terminal (usually is a smart phone) that supports password-based authentication, cryptographic computation and biometric extraction, a cloud center with powerful computation and storage capabilities, a gateway with relatively limited resource and lots of resource-constrained sensor nodes. Besides, four basic phases are included: registration, login, authentication and password change phase. Usually, the dynamics node addition phase and re-registration phase are provided to achieve the repairability and extensibility of the authentication scheme. At the beginning, both the cloud center CS and the gateway GWN_k will get their long term secret key x and x_{G_k}, respectively. In the registration phase, the user U_i and the gateway GWN_k need to submit their personal information to register to the cloud center CS, and the sensor node S_j needs to submit its identity to GWN_k. This phase takes place in a secure channel $\Longrightarrow$. After the registration phase, U_i has some sensitive parameters stored in her smart phone and shares a secret parameter with CS; GWN_k and CS will form a secret shared key X_{G_k} (also called as GWN_k's secret key); S_j and GWN_k will share a secret key X_{S_j} (also called as S_j's secret key). Then the user U_i will choose a target sensor node S_j and initiates a login request to CS via a public channel $\rightarrow$. Then U_i and S_j will authenticate each other and negotiate a session key to protect their subsequent communications. Furthermore, the user can update her password in the password change phase, and re-register to the system in re-registration phase if her account is blocked.

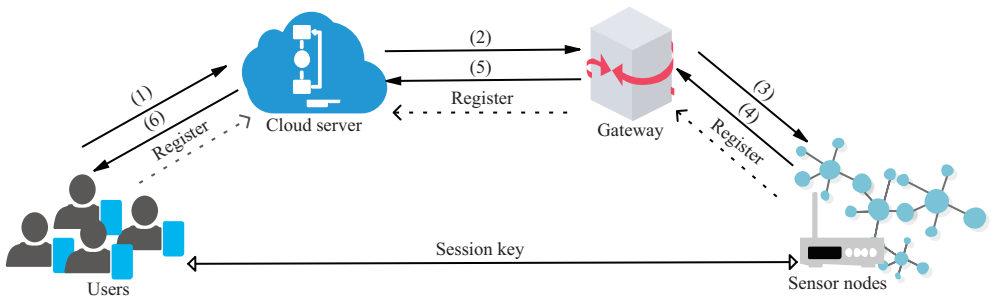

Figure 2 (Color online) Authentication architecture of industry 4.0.

Table 2 Evaluation criteria [6]

Short term	Definition
C1: No password verifier-table	CS and GWN_k should not store any password-related data
C2: Password friendly*	U_i is able to choose her password and change it locally
C3: Sound repairability	The dynamic node addition and user re-registration phase are provided
C4: Key agreement	The sensor node and the user should share a same session key for future communication after the authentication
C5: No clock synchronization	The participants do not have to keep a same time clock
C6: No password exposure	CS cannot get/compute user's password
C7: Mutual authentication	All the participants should authenticate each other
C8: User anonymity	User's identity should not be exposed to the adversary
C9: No smart card loss attack	The adversary cannot conduct an attack by making use of the data stored in smart cards/smart terminals
C10: Resistance to known attacks	The adversary without smart card/smart terminals cannot carry out known attacks in Table 1 of [6], such as node capture attack
C11: Forward secrecy	The adversary is impossible to get the previous session keys, even getting the long term secret key

2.2 Adversary model and evaluation criteria

A well-defined evaluation criteria and adversary model put authentication schemes on a common spectrum, which makes it possible to assess the schemes fairly and comprehensively. Recently, Wang et al. [6] give a systematic and widely-accepted adversary model and evaluation criteria for WSNs. According to Wang et al.'s adversary model [6] which is developed from Dolev-Yao [26], we form ours as follows:

(D1) $\mathcal{A}$ can enumerate all user-chosen passwords and identities within polynomial time; or know the victim's identity when assessing the security of the scheme.

(D2) $\mathcal{A}$ can intercept, modify, eavesdrop all the messages that are transmitted in a public channel among the users, cloud center, gateway and sensor nodes. Note that, in registration phase, the channel is supposed to be secure.

(D3) $\mathcal{A}$ can break any two of the three factors (password, smart card/smart terminal and biometrics).

(D4) $\mathcal{A}$ can obtain the previous session keys between users and sensor nodes.

(D5) $\mathcal{A}$ can get the secret key of the cloud center, the gateway or the sensor nodes when considering forward secrecy.

(D6) $\mathcal{A}$ can capture the data stored in some sensor nodes.

(D7) $\mathcal{A}$ can register as a legitimate user, gateway or sensor node. In other words, these three participants try to get additional data or privileges, but they still follow the protocol.

According to Wang et al.'s twelve independent evaluation criteria [6], we form ours as shown in Table 2. Among these criteria, "forward secrecy" and "resistance to smart card loss attack" are the most two difficult criteria to be achieved. Smart card loss attack is the most prevalent [17]. Forward secrecy is significant to high security requirement applications, it can improve the system's resistance to the final attack. To resist smart card loss attacks, the "fuzzy-verifier" techniques and public-key techniques are necessary [10]. Yet, when applying public-key techniques, most of schemes require gateways to conduct at least one modular exponentiation or point multiplication operation, which can be a computing bottleneck with the number of concurrent sessions increasing. To achieve forward secrecy, at least two modular

exponentiation or point multiplication at sensor nodes is indispensable [20]. However, owing to the resource-constrained nature of sensor nodes, computational expensive cryptographic primitive on sensor nodes is not practical. Thus, how to achieve forward secrecy efficiently is still an open issue. Most schemes either ignore the resource limitation nature of sensor nodes or attempt to only use lightweight operations (these schemes usually cannot provide forward secrecy). In this paper, taking the advantage of the imbalanced computational nature of RSA cryptosystem, we try to achieve forward secrecy efficiently.

3 Formal security model

Provable security theory has been one of the most effective ways of analyzing the security of complex protocols. Based on [9,10], this section defines the security model to formally capture adversary's capabilities and possible attack processes.

Players. In our four-party-protocol $\mathcal{P}$, the participants involved are: a user $U \in \mathsf{User}$, a cloud center $\mathrm{CS} \in \mathsf{CloudServer}$, a gateway $G \in \mathsf{Gateway}$ and a sensor node $S \in \mathsf{Sensor\ node}$. When specific to a conversation, U is instantiated as U_i, similarly, $G \Rightarrow G_k$, $\mathrm{CS} \Rightarrow \mathrm{CS}_m$ and $S \Rightarrow S_j$, $i, j, k, m \in \mathbb{Z}$. Let I be any kind of instances, and I^s be the s-th instance of I.

Long-lived keys. Before the authentication, the four participants need to build their basis of trust firstly in registration phase. Generally, the cloud center owns a long-term secret key $\mathrm{cpri}_{\mathrm{key}}$. To a scheme using a public-key algorithm, there is also a corresponding public key $\mathrm{pub}_{\mathrm{key}}$. Then the gateways have their secret key $\mathrm{gwn}_{\mathrm{key}}$ stored in a database; users will also get their unique secret key $\mathrm{ue}_{\mathrm{key}}$ which is not stored at the user side, but can be computed by the user's passwords and parameters in a smart terminal. Note that, in our four-party system, the gateway also owns a long-lived key $\mathrm{gpri}_{\mathrm{key}}$, and it will distribute the sensor nodes their unique secret key $\mathrm{sen}_{\mathrm{key}}$. The cloud center and the gateway do not directly store $\{\mathrm{sen}_{\mathrm{key}}, \mathrm{ue}_{\mathrm{key}}\}$ and $\mathrm{gwn}_{\mathrm{key}}$, respectively, but the parameters can derive them.

Queries. $\mathcal{A}$ can only interact with the protocol participants via oracle queries. The queries can capture the adversary capabilities in a real attack and are defined as follows.

Execute$(U_i^r, \mathrm{CS}_m^d, G_k^s, S_j^t)$: This query outputs the transcript among the four instances. It models a passive attack (usually is eavesdropping).

Send(I, I_i^s, m): This query means that I sends the message m to instance I_i^s, and I gets a response from I_i^s according to $\mathcal{P}$. But if m is invalid, this query is ignored.

Reveal(I_j^s): It models the known-key attack where I can be U or S. If I^s has negotiated a session key, then the oracle outputs the key, otherwise $\perp$.

Corrupt(I, a): It models $\mathcal{A}$'s corruption capability and is defined as follows.

(a) When $I == U_i$, $\mathcal{A}$ can break any two of the three factors.

- If $a = 1$, $\mathcal{A}$ gets the password PW_i and the biometric fng_i, written as [output$(\mathrm{PW}_i, \mathrm{fng}_i)$];
- If $a = 2$, $\mathcal{A}$ gets the password and the data in smart terminal, written as [output$(\mathrm{SC}_i, \mathrm{PW}_i)$];
- If $a = 3$, $\mathcal{A}$ gets the biometric and the data in smart terminal, written as [output$(\mathrm{fng}_i, \mathrm{SC}_i)$].

(b) When $I == \mathrm{CS}_m$, output secret key $\{x, y\}$ and verifier table $\{\mathrm{ID}_i, x_i, \mathrm{SUM}\}$, denoted as [output($x$, y, verifier table)].

(c) When $I == G_k$, output secret key x_{G_k} and X_{G_k}, denoted as [output$(x_{G_k}, X_{G_k}$, verifier table)].

(d) When $I == S_j$, output the shared secret key X_{S_j} of S_j, written as [output(X_{S_j})].

Test(I_j^s): It does not aim at modeling the attack, but testing the semantic security of session key. If I_j^s has not yet built a session key or I_j^s is not fresh, or Test(I_j^s) has been queried before, then the answer is $\perp$; otherwise, the oracle will flip a coin b. If $b == 1$, it outputs the real session key; if $b == 0$, it outputs a random number with the same size.

Partnering. The session identifier sid is used to define partnering. Suppose there are two instances U_i^s and S_j^r, then two instances U_i^s and S_j^r are partnered only if the following conditions are satisfied:

- They are accepted.
- Sid U_i^s == sid S_j^r where sid U_i^s or sid S_j^r denotes the session identifier of U_i^s or S_j^r.
- Pid U_i^s == S_j^r, pid S_j^r == U_i^s, where pid U_i^s denotes the session identifier of U_i^s's partner.

Freshness. Generally, the adversary's goal is to guess the value of b in Test(I_j^s) to break the semantic security. However, suppose a situation where $\mathcal{A}$ queries Reveal(I_j^s) and Test(I_j^s) at the same time, and then $\mathcal{A}$ of course can judge the value of b. Note that in this situation, $\mathcal{A}$ actually does not break the semantic security. Therefore, it is necessary to make some restrictions to allow $\mathcal{A}$ to query Test(I_j^s) reasonably. So we define the notion of freshness. A fresh instance I should satisfy the following.

- I has accepted and gotten the session key.
- $\mathcal{A}$ has not yet asked Reveal query for I and its partner.
- Corrupt(U, a) is queried no more than one time.
- Corrupt(CS,1) is not asked. Or though it is asked, $\mathcal{A}$ does not query Send-query after that.
- Corrupt$(S, 1)$ is not asked. Or though it is asked, $\mathcal{A}$ does not query Send-query after that.
- Corrupt$(G, 1)$ is not asked. Or though it is asked, $\mathcal{A}$ does not query Send-query after that.

Correctness. If U_i, CS_m, S_j and G_k are partnered and accepted, U_i and S_j will share a session key.

Semantic security. Consider a protocol $\mathcal{P}$, there is an adversary $\mathcal{A}$ who has asked Execute, Send, Reveal queries with a polynomial number, as well as a Test-query to fresh instances. $\mathcal{A}$ tries to against $\mathcal{P}$ via guessing the value of b. Let Succ be the event that $\mathcal{A}$ guessed b correctly. The advantage of $\mathcal{A}$ in breaking semantic security of $\mathcal{P}$ is $\mathrm{Adv}^{\text{sfs-ake}}_{\mathcal{P},\mathcal{D}}\mathcal{A} = 2\mathrm{Pr}[\mathrm{Succ}\mathcal{A}] - 1$.

We say a password-authenticated protocol is semantically secure, when in the polynomial time, the advantage of an adversary $\mathcal{A}$ who makes at most q_{send} active attacks, satisfies

$$\mathrm{Adv}^{\text{ake}}_{\mathcal{P},\mathcal{D}}(\mathcal{A}) < C' q^{s'}_{\text{send}} + \varepsilon(\ell),$$

where $\mathcal{D}$ is the password space whose frequency distribution follows a Zipf's law [27], C' and s' are Zipf parameters [10], $\varepsilon(\cdot)$ is a negligible function, and l is a system security parameter.

4 The proposed scheme

This section introduces an efficient and secure three-factor remote user authentication scheme which is suitable for a cloud-aided industry 4.0 environment (see Figure 3). We exploit the imbalanced computational nature of the RSA cryptosystem to provide forward secrecy without expensive computation cost on the sensor nodes. In the RSA algorithm, the public key e is recommended to be a small prime (e.g., $2^{16} + 1$), the operation $m^e \bmod n$ costs less than the operation $g^a \bmod p$ in ElGamal cryptosystems (or aP operations in ECC cryptosystems) where a has a large bit length to ensure security. The RSA cryptosystem is shown as follows.

- Key generation. Select two distinct prime numbers (p, q), compute $n = p \cdot q$, $r = (p-1)(q-1)$, $d \equiv e^{-1} \bmod r$, where e is an integer satisfying $1 < e < r$ (e and r are coprime). The public key is (n, e), and the private key is (n, d).
- Encryption. $c \equiv m^e \bmod n$.
- Decryption. $c^d \equiv (m^e)^d \equiv m \bmod n$.

4.1 Sensor node and gateway registration phase

Since we regard the user, the cloud center and the gateway are three different stakeholders. To better security, we do not put the eggs in a basket. In other words, let the cloud center have two long term secret keys x and y. x is used to compute secret parameters between gateways and the cloud center; and y is to compute secret parameters between users and the cloud center. Thus, once one of the keys is exposed, only one of the stakeholders (the user or the gateway) will be affected. Besides, the gateway also has two secret keys, one is X_{G_k} $(= h(\mathrm{GID}_k||x))$, the other is x_{G_k}. X_{G_k} is shared with the cloud center. Note that, each gateway has its own unique shared secrecy key. Therefore, the security of the gateways will not be affected by each other. x_{G_k} is applied to the network that consists of the gateway and the sensor nodes, which guarantees the independence of the wireless sensor network. In our scheme, the users and the gateways need to register to the cloud center, and the sensor nodes need to register to the gateways. Note that, we build the scheme on an elliptic curve E. To an elliptic curve E over prime finite field F_p, we have $y^2 = x^3 + ax + b \bmod p$, where $a, b \in F_p$, and $4a^3 + 27b^2 \neq 0 \bmod p$. Assume P is a point on F_p, aP is the result of computing P times a. The cloud center select x and y as its long term secret key, the public key is $Y = y \cdot P$. In addition, the hash function $h : 0,1^* \to 0,1^l$ with the input $s \in 0,1^*$ and output $t \in 0,1^l$, should satisfy the following properties: (1) given s, it can compute t such that $t = h(s)$ within polynomial time; (2) given t, it should be computationally infeasible to find the input s such that $t = h(s)$; (3) given s, it should be computationally infeasible to find s' $(s' \neq s)$ such that $h(s') = h(s)$.

The sensor node can register to the gateway according to the following steps.

Step 1. $S_j \Longrightarrow \mathrm{GWN}_k$: registration request.

Step 2. $\mathrm{GWN}_k \Longrightarrow S_j$: $\{\mathrm{SID}_j, X_{S_j} = h(\mathrm{SID}_j||x_{G_k})\}$, where x_{G_k} is GWN_k's secret key.

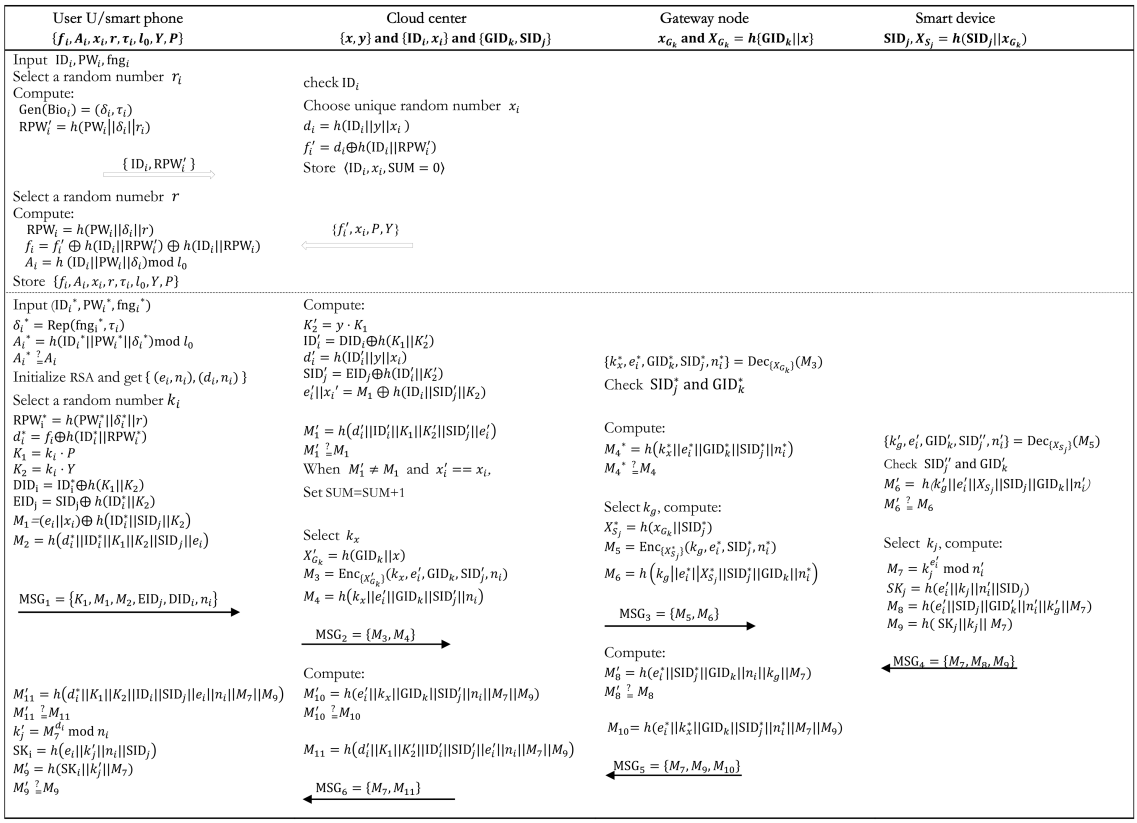

Figure 3 The proposed scheme.

Step 3. S_j stores X_{S_j}.

The gateway can register to the cloud center as the steps below.

Step 1. $\text{GWN}_k \Longrightarrow \text{CS}$: registration request and the identity of the sensor nodes that are connected to it.

Step 2. $\text{CS} \Longrightarrow \text{GWN}_k$: $\{\text{GID}_k, X_{G_k}(= h(\text{GID}_k||x))\}$, the cloud center stores the identity of the gateway and its corresponding sensor nodes as $\{\text{GID}_k, \text{SID}_j\}$.

Step 3. GWN_k keeps X_{G_k}.

4.2 User registration phase

Step 1. $U_i \Longrightarrow \text{CS}$: $\{\text{ID}_i, \text{RPW}'_i\}$. U_i inputs her selected identity ID_i and password PW_i and biometric fng_i, and then submits $\{\text{ID}_i, \text{PW}_i, \text{fng}_i\}$ to the smart phone. Then it selects a random number r_i and computes: $\text{Gen}(\text{fng}_i) = (\delta_i, \tau_i)$, $\text{RPW}'_i = h(\text{PW}_i||\delta_i||r_i)$.

Step 2. $\text{CS} \Longrightarrow U_i$: $\{f'_i, x_i, P, Y\}$. The cloud center first checks the valid of the identity ID_i, then picks a unique random number x_i for U_i, and computes $d_i = h(\text{ID}_i||y||x_i)$, $f'_i = d_i \oplus h(\text{RPW}'_i||\text{ID}_i)$, then stores $\{\text{ID}_i, x_i, \text{SUM} = 0\}$ in the database.

Step 3. After getting the response, the smart terminal computes: $\text{RPW}_i = h(\text{PW}_i||\delta_i||r)$ using a new random number for stopping insider attack [28], $f_i = f'_i \oplus h(\text{ID}_i||RPW_i) \oplus h(\text{ID}_i||\text{RPW}_i)$, $A_i = h(\text{ID}_i)||h(\text{PW}_i)||h(\delta_i)) \bmod l_0$ where $l_0 = 2^8$, then keeps $\{f_i, A_i, A_i \oplus x_i, r, \tau_i, Y, P, l_0\}$.

Here we apply Wang et al.'s method [10] to deal with the conflict between detecting the wrong input timely and resisting offline dictionary attacks. More specifically, let the verifier A_i be computed as $h(\text{ID}_i||\text{PW}_i||\delta_i) \bmod l_0$. In this way, even an adversary acquires the victim's biometric and finds a pair of identity and password $\{\text{ID}_i^*, \text{PW}_i^*\}$ that satisfies the equation $h(\text{ID}_i^*||\text{PW}_i^*||\delta_i) \bmod l_0$, she still has to further check its correctness online because there are $\frac{|\mathcal{D}_{\text{id}} \times \mathcal{D}_{\text{pw}}|}{l_0} \approx 2^{32}$ candidate pairs that satisfy the equation when $l_0 = 2^8$. However, the online guessing attack can be prevented by the parameter "SUM" which is used to count the failure times of the authentication. Once its value exceeds the preset value (such as 10), then the user's account will be suspended. Accordingly, our scheme detects the wrong input timely, and also resist offline dictionary attack caused by the verifier A_i.

4.3 Login phase

Step 1. $U_i \longrightarrow$ CS: $\{K_1, M_1, M_2, \mathrm{EID}_j, \mathrm{DID}_i, n_i\}$. U_i enters $\{\mathrm{ID}_i^*, \mathrm{PW}_i^*, \mathrm{fng}_i^*\}$, and then the smart terminal computes: $\delta_i^* = \mathrm{Rep}(\mathrm{fng}_i^*, \tau_i)$, $h(\mathrm{ID}_i^*||\ \mathrm{PW}_i^*||\delta_i) \bmod l_0$, then checks whether $A_i^* \stackrel{?}{=} A_i$ to authenticate the user.

If equation does not hold, the smart terminal rejects the request. Otherwise, it initializes a pair of RSA parameter $\{(e_i, n_i), (d_i, n_i)\}$, chooses a random number k_i, computes: $\mathrm{RPW}_i^* = h(\mathrm{PW}_i^*||\delta_i^*||r)$, $d_i^* = f_i \oplus h(\mathrm{ID}_i^*||\mathrm{RPW}_i^*)$, $K_1 = k_i \cdot P$, $K_2 = r_i \cdot Y$, $\mathrm{DID}_i = \mathrm{ID}_i^* \oplus h(K_1||K_2)$, $\mathrm{EID}_j = \mathrm{SID}_j \oplus h(\mathrm{ID}_i^*||K_2)$, $M_1 = e_i||x_i \oplus h(\mathrm{ID}_i^*||\mathrm{SID}_j||K_2)$, $M_2 = h(d_i^*||\mathrm{ID}_i^*||K_1\ ||K_2||\mathrm{SID}_j||e_i)$.

4.4 Authentication phase

Step 1. CS $\longrightarrow$ GWN$_k$: $\{M_3, M_4\}$. Once receiving the login request, the cloud center computes $K_2' = y \cdot K_1$, $\mathrm{ID}_i' = \mathrm{DID}_i \oplus h(K_1||K_2')$, retrieves x_i from the database, then computes $d_i' = h(\mathrm{ID}_i'||y||x_i)$, $\mathrm{SID}_j' = \mathrm{EID}_j \oplus h(\mathrm{ID}_i'||K_2')$, $e_i'||x_i' = M_1 \oplus h(\mathrm{ID}_i'||\mathrm{SID}_j'||K_2)$, $M_1' = h(d_i'||\mathrm{ID}_i'||K_1||K_2'\ ||\mathrm{SID}_j'||e_i')$.

If $M_1' \neq M_1$, the cloud center rejects the request and checks the value of x_i' and x_i. If $x_i' == x_i$, and the cloud center sets SUM = SUM + 1. Once SUM exceeds a preset value (such as 10), the cloud center will suspend U_i's account.

If $M_1' == M_1$, the cloud center thinks the request is valid, then selects a random number k_x, and determines the corresponding gateway GWN$_k$, then computes $X_{G_k}' = h(\mathrm{GID}_k||x)$, $M_3 = \mathrm{Enc}_{X_{G_k}'}(k_x, e_i', \mathrm{GID}_k, \mathrm{SID}_j', n_i)$, $M_4 = h(k_x||e_i'||\mathrm{GID}_k||\mathrm{SID}_j'||n_i)$.

Step 2. GWN$_k \longrightarrow S_j$: $\{M_5, M_6\}$. The gateway decrypts M_3 with X_{G_k}' to get $\{k_x^*, e_i^*, \mathrm{GID}_k^*, \mathrm{SID}_j^*, n_i^*\}$. After checking the valid of $\{\mathrm{GID}_k^*, \mathrm{SID}_j^*\}$, the cloud center computes $M_4^* = h(k_x^*||e_i^*||\mathrm{GID}_k^*||\mathrm{SID}_j^*||n_i^*)$.

If $M_4^* \neq M_4$, GWN$_k$ rejects the session. Otherwise, it selects a random number k_g, and computes $X_{S_j}^* = h(x_{G_k}||\mathrm{SID}_j^*)$, $M_5 = \mathrm{Enc}_{X_{S_j}^*}(k_g, e_i^*, \mathrm{SID}_j^*, n_i^*)$, $M_6 = h(k_g||e_i^*||X_{S_j}^*||\mathrm{SID}_j^*||\mathrm{GID}_k||n_i^*)$.

Step 3. $S_j \longrightarrow$ GWN$_k$: $\{M_7, M_8, M_9\}$. The sensor node decrypts M_5 using its private key X_{S_j}, gets $\{k_g', e_i', \mathrm{GID}_k', \mathrm{SID}_j'', n_i'\}$, checks the valid of GID_k' and SID_j''. Then, the sensor node computes $M_6' = h(k_g'||e_i'\ ||X_{S_j}||\mathrm{SID}_j||\mathrm{GID}_k||n_i')$.

If $M_6' \neq M_6$, the sensor node rejects the access. Otherwise, it computes $M = k_j^{e_i'} \bmod n_i'$, $\mathrm{SK}_j = h(e_i'||k_j||n_i'||\mathrm{SID}_j)$, $M_8 = h(e_i'||\mathrm{SID}_j||\mathrm{GID}_k'||n_i'||k_g'||M_7)$, $M_9 = h(\mathrm{SK}_j||k_j||M_7)$.

Step 4. GWN$_k \longrightarrow$ CS: $\{M_7, M_9, M_{10}\}$. The gateway computes $M_8' = h(e_i^*||\mathrm{SID}_j^*||\mathrm{GID}_k||n_i\ ||k_g||M_7)$, compares M_8' with M_8 to authenticate S_j. If the equation does not hold, terminates the session, otherwise computes $M_{10} = h(e_i^*||k_x^*||\mathrm{GID}_k||\mathrm{SID}_j^*||n_i^*||M_7||M_9)$.

Step 5. CS $\longrightarrow U_i$: $\{M_7, M_{11}\}$. The cloud center computes $M_{10}' = h(e_i'||k_x||\mathrm{GID}_k||\mathrm{SID}_j'\ ||n_i||M_7||M_9)$. If $M_{10}' \neq M_{10}$, ends the session; Otherwise, computes $M_{11} = h(d_i'||K_1||K_2'||\mathrm{ID}_i'||\mathrm{SID}_j'||e_i'\ ||n_i||M_7||M_9)$.

Step 6. The smart terminal computes $M_{11}' = h(d_i^*||K_1||K_2||\mathrm{ID}_i||\ \mathrm{SID}_j||e_i||n_i||M_7||M_9)$. If $M_{11}' \neq M_{11}$, the smart terminal terminates the session and the authentication fails. According to the RSA algorithm, when the ciphertext is c, the plaintext is m and the pair of public/private key is $((e, n), (d, n))$, it has $c^d \bmod n = (m^e)^d \bmod n$. Therefore, the user can use the private key d_i to decrypt M_7, and get k_j' by computing $M_7^{d_i} \bmod n_i$, after that, the user computes the session key $\mathrm{SK}_i = h(e_i||\ k_j'||n_i||\mathrm{SID}_j)$, $M_9' = h(\mathrm{SK}_i||k_j'||M_7)$. If $M_9' == M_9$, it means that the user and the sensor node have shared a same session key and the authentication is finished successfully. Note that, all the participants should delete the RSA parameters. Otherwise, the user ends the session and rejects the session key.

4.5 Password change phase

Step 1. $U_i \rightarrow$ smart terminal: $\{\mathrm{ID}_i^*, \mathrm{PW}_i^*, \mathrm{fng}_i^*, \mathrm{PW}_i^{\mathrm{new}}\}$.

Step 2. The smart terminal will firstly verify the identity of the user: computes $\delta_i^* = \mathrm{Rep}(\mathrm{fng}_i^*, \tau_i)$, $A_i^* = h(\mathrm{ID}_i^*\ ||\mathrm{PW}_i^*||\delta_i^*) \bmod l_0$, compares A_i^* with A_i.

If $A_i^* \neq A_i$, the terminal ends the session; otherwise it computes $\mathrm{RPW}_i^* = h(\mathrm{PW}_i^*||\delta_i^*||r)$, $d_i^* = f_i \oplus h(\mathrm{ID}_i||\mathrm{RPW}_i^*)$, $\mathrm{RPW}_i^{\mathrm{new}} = h(\mathrm{PW}_i^{\mathrm{new}}||\delta_i^*||r)$, $A_i^* = h(h(\mathrm{ID}_i^*||\ h(\mathrm{PW}_i^{\mathrm{new}})||h(\delta_i^*)) \bmod l_0$, $f_i^{\mathrm{new}} = d_i^* \oplus h(\mathrm{ID}_i^*||\ \mathrm{RPW}_i^{\mathrm{new}})$, then replaces $\{f_i, A_i\}$ with $\{f_i^{\mathrm{new}}, A_i^{\mathrm{new}}\}$.

4.6 Re-registration phase

According to our scheme, once the value of SUM exceeds the preset value, the user's account will be suspended. Therefore, it is necessary to provide a way for the users to activate their accounts. Usually, people's identities are personal-related values, and it is not convenient to change the identity. Our scheme allows the user to re-register using the previous identity to activate their account as follows.

Step 1. The $U_i \rightarrow$ smart terminal: $\{\mathrm{ID}_i^*, \mathrm{PW}_i^*, \mathrm{fng}_i^*, \mathrm{PW}_i^{\mathrm{new}}$, revoke-requst$\}$. The user initiates the re-registration request to the smart terminal.

Step 2. $U_i \Longrightarrow$ CS: $\{\mathrm{ID}_i^*, \mathrm{RPW}_i', x_i$, revoke-requst$\}$. The smart terminal will firstly verify the identity of the user: computes $\delta_i^* = \mathrm{Rep}(\mathrm{fng}_i^*, \tau_i)$, $A_i^* = h(\mathrm{ID}_i^* \,||\mathrm{PW}_i^*||\delta_i^*) \bmod l_0$, compares A_i^* with A_i.

If $A_i^* \neq A_i$, rejects; otherwise, selects a new random number r_i, computes $\mathrm{RPW}_i' = h(\mathrm{PW}_i^{\mathrm{new}}||\delta_i^*||r_i)$.

Step 3. CS $\Longrightarrow U_i$: $\{f_i^{\mathrm{new}}, x_i^{\mathrm{new}}, P, Y\}$. The cloud center CS first checks the valid of ID_i and x_i: if any of them are not in the database, CS rejects the request; otherwise, CS picks the new unique random number x_i^{new}, computes $d_i = h(\mathrm{ID}_i^*||y||x_i^{\mathrm{new}})$, $f_i^{\mathrm{new}} = h(\mathrm{RPW}_i'||\mathrm{ID}_i^*) \oplus d_i^{\mathrm{new}}$, and stores $\{\mathrm{ID}_i^*, x_i^{\mathrm{new}}, \mathrm{SUM} = 0\}$.

Step 4. The smart terminal selects a random number r^{new}, computes $\mathrm{RPW}_i = h(\mathrm{PW}_i||\delta_i||r^{\mathrm{new}})$, $A_i^{\mathrm{new}} = h(\mathrm{ID}_i^* \,||\mathrm{PW}_i^{\mathrm{new}}||\delta_i^*) \bmod l_0$, $f_i^{\mathrm{new}} = h(\mathrm{ID}_i^*||\mathrm{RPW}_i) \oplus h(\mathrm{ID}_i^*||\mathrm{RPW}_i') \oplus f_i'$, keeps $\{f_i^{\mathrm{new}}, A_i^{\mathrm{new}}, x_i^{\mathrm{new}}, r, \tau_i, P, Y, l_0\}$.

4.7 Dynamics node addition phase

When a new sensor node wants to join the network, it can register to the gateway as the sensor node registration phase in Subsection 4.1. After it successfully registers, the gateway should inform the cloud center of the identity (SID_j) of the new sensor node as follows:

Step 1. $\mathrm{GWN}_k \Longrightarrow$ CS: $\{\mathrm{SID}_j, h(\mathrm{SID}_j||X_{G_k})\}$.

Step 2. The cloud center computes $X'_{G_k} = h(\mathrm{GID}_k||x)$, then compares $h(\mathrm{SID}_j||X'_{G_k})$ with $h(\mathrm{SID}_j||X_{G_k})$. If they are equal, the cloud center adds SID_j in the database, and gives a positive response to the gateway. Otherwise, the cloud center rejects the request.

5 Formal security analysis

In this section, the security of the proposed scheme is analyzed under the model defined in Section 3. When the simulation begins, the algorithm `Init` first runs an algorithm $\mathcal{G}$ to generate a q-order subgroup P on an elliptic curve E/F_p over the finite field F_p (where p and q are two large primes, and $|p| = \ell$ (security parameter)), several collision-resistant hash functions $\mathcal{H}_i : \{0,1\}^* \rightarrow \{0,1\}^{l_i}$ $(i = 0, 1, 2, 3)$, the encryption and decryption function $\mathcal{E}(\cdot)/\mathcal{D}(\cdot) : \{0,1\}^* \rightarrow \{0,1\}^l$, and two long-term private keys and a public key $(x/y, yP)$. In our proof, let the adversary $\mathcal{A}$ interact with honest participants through querying oracles and controlling the simulator. Eventually, with all information from the oracles, $\mathcal{A}$ tries to determine whether a given number is a valid key to break the protocol. Furthermore, before the security analysis, we recall the following hypothesis.

Computational Diffie-Hellman (CDH) problem. Let g be the generator of $\mathcal{G}$, given g^x and g^y, the advantage for $\mathcal{A}$ to compute g^{xy} in polynomial time t is $\mathrm{Adv}^{\mathrm{CDH}}(t) \leqslant \varepsilon$, where ε is an ignorable small number.

Elliptic curve computational Diffie-Hellman (ECDH) problem. Given aP and bP in $\mathcal{G}$, the advantage for $\mathcal{A}$ to compute abP within polynomial time t is $\mathrm{Adv}^{\mathrm{ECDH}}(t) \leqslant \varepsilon$.

Theorem 1. Let $\mathcal{P}$ be the protocol we proposed, $|\mathcal{D}|$ be the space of password, and l be the security length. Then an adversary against the semantic security of $\mathcal{P}$ makes q_s times Send$(\cdot)$-queries, q_e times Execute$(\cdot)$-queries and q_h times hash$(\cdot)$-queries with the polynomial time t, and then her advantage to break the semantic security of $\mathcal{P}$ is

$$\begin{aligned}\mathrm{Adv}_{\mathcal{P}}^{(\mathrm{ake})}(\mathcal{A}) \leqslant & \frac{(q_s+q_e)^2}{2p} + \frac{2q_h^2 + 2(q_s+q_e)^2 + (2q_s+q_h)^2 + 4q_s^2}{2^l} \\ & + C' q_{\mathrm{send}}^{s'} + q_h(1 + 2(q_s+q_e)^2)\left(\mathrm{Adv}_p^{\mathrm{ECDH}}(t') + \mathrm{Adv}_p^{\mathrm{CDH}}(t')\right),\end{aligned}$$

where C' and s' are Zipf parameters [29], T_m is time for scaler multiplication operation in $\mathcal{G}$, and $t' \leqslant t + (q_s + q_e + 1) \cdot T_m$.

Table 3 The running time of RSA initiation algorithm (ms)

Experimental platform	RSA ($\|n\|$=1024)	RSA ($\|n\|$=2048)
A smart phone released in 2016 with MSM8998 CPU, 6GB LPDDR4 memory and H2OS 1.4 system (based on Android 6.0)	21	103
Common laptop with Intel(R) Xeon(R), CPU E5-2640 v4 @ 2.40 GHz and 512 G	13	84
Common laptop with Intel(R) Xeon(R) Gold 6254, CPU @3.10 GHz and 256 G	11	69

Proof. We prove Theorem 1 via a sequence of games from a real attack game G_0 to game G_6. The advantage between the adversary of these games is gradually decreasing to zero. The detailed proof can be found in Appendix A.

6 Performance analysis

In this section, we compare our scheme with the state-of-the-art relevant schemes in terms of the security, computation cost, communication and storage cost, as shown in Tables 4–6 [3,23,30–37] respectively. Computation cost refers to the calculation time required for each participant to complete the authentication. Communication cost refers to the bit size of the messages transmitted among participants. Storage cost refers to the bit size of parameters stored in each participants. We assume that the bit size of l_0 is 32 bits, the random number, identity, timestamp are 128 bits, the hash output and ECC-based parameters are 160 bits, and the secret key of symmetric-cryptosystem-based and the extended-chaotic-maps-cryptosystem-based schemes are 1024 bits. Let T_C, T_P, T_B, T_H, T_S denote the operation time of Chebyshev chaotic-map, elliptic curve point multiplication, biometric fuzzy extracting, hash and symmetric encryption, respectively. Note that we ignore some lightweight operations such as XOR and $\|$. Based on [10], when using Intel(R) i3-530 2.93-GHz and the standard cryptographic library MIRACL, the running time for the modular exponentiation, the ECC point multiplication, symmetric cryptography (AES-128) and hash function (SHA-1) is 1.169 ms, 0.508 ms, 0.541 μs and 0.693 μs, respectively. Note that, we use this environment to approximate the use side, gateway and sensor nodes. Although, the running time of these operations on three participants are different, it does not affect the results of the comparisons and this approach has been accepted by the community, and some recent typical studies include [3, 30, 33, 36]. Furthermore, based on [30], we can compute the running time of the big-exponent modular exponentiation $T_{\rm be} \approx 1.169 \times 8 = 9.352$ ms, the small-exponent modular exponentiation $T_{\rm se} \approx 17/1024 \times T_{\rm be} = 0.156$ ms for RSA $|n| = 1024$ and $e = 2^{16} + 1$. Based on [33, 36], we have $T_C \approx T_P \approx 0.508$ ms.

As shown in Table 3, to test the feasibility and efficiency of the RSA initialization algorithm, we write the initiation algorithm using the Java language and conduct it on different environments. The result indicates that the running time of the RSA initialization on a smart phone released in 2016 which includes MSM8998 CPU, and 6 GB LPDDR4 memory along with H2OS 1.4 system (based on Android 6.0), is approximately 21 ms when the length of the private key of RSA is 1024 bits. Furthermore, note that the addition of biometric factors will inevitably increase the computation costs on the user side, but it increases the difficulty for adversaries to attack authentication schemes.

From the security perspective as shown in Table 4, our scheme is the only one that satisfies all eleven widely-accepted criteria [6]. Among the remaining schemes, Wang et al. [30] and Li et al. [31] perform the best; they achieve ten criteria and cannot meet the indicator C6 "No password exposure". The schemes of Sharif et al. [32], Wu et al. [34], Amin et al. [35] and Wazid et al. [36] only use symmetric cryptographic algorithms and all cannot achieve forward secrecy.

From the computation cost perspective as shown in Table 5, our scheme has the lowest computation cost on gateways (0.005 ms), and our computation cost (0.159 ms) on sensor nodes is even comparable to these schemes using only symmetric cryptographic algorithms (Sharif et al. [32] 0.003 ms, Wu et al. [34] 0.003 ms, Amin et al. [35] 0.004 ms, Wazid et al. [36] 0.005 ms), and is much lower than these schemes that also use asymmetric cryptographic algorithms (Wang et al. [30] 0.159 ms, Li et al. [31] 1.019 ms, Srinivas et al. [38] 1.020 ms, Li et al. [3] 1.019 ms, Park et al. [37] 1.019 ms, Reddy et al. [23] 1.019 ms). However, our computation cost on the user side is higher than most schemes. Yet, since the user usually owns a powerful smart terminal, such as a smart phone, conducting these cryptographic algorithms is not a bottleneck. Besides, the computation cost on the user side only affects the user experience. Specifically, to initiate an access request, in our scheme, the user needs to cost nearly 1.02 s when assuming that the

Table 4 Security comparison among relevant authentication schemes

Protocol	Year	Ref.	Evaluation criteria										
			C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
Wang et al.	2019	[30]	√	√	√	√	√	×	√	√	√	√	√
Li et al.	2019	[31]	√	√	√	√	√	×	√	√	√	√	√
Sharif et al.	2019	[32]	√	√	√	√	×	√	√	√	×	×	×
Srinivas et al.	2018	[33]	√	√	√	√	×	×	√	√	×	×	√
Li et al.	2018	[3]	√	√	√	√	√	×	√	√	×	×	√
Wu et al.	2018	[34]	√	×	√	√	√	×	√	√	×	×	×
Amin et al.	2018	[35]	√	√	√	√	×	×	√	√	×	×	×
Wazid et al.	2017	[36]	√	√	√	√	×	×	√	√	×	×	×
Park et al.	2016	[37]	√	√	√	√	×	×	√	×	×	×	√
Reddy et al.	2016	[23]	√	×	√	√	√	×	√	√	√	×	√
Our scheme	–		√	√	√	√	√	√	√	√	√	√	√

Table 5 Computation costs among relevant authentication schemes

Protocol	Year	Ref.	Login (ms)	Authentication (ms)		
			User	User	Gateway	Sensor node
Wang et al.	2019	[30]	$T_{se} + 8T_H + T_I \approx 21.162$	$T_{be} + 3T_H \approx 9.354$	$T_{be} + T_S + 6T_H \approx 9.356$	$T_{se} + T_S + 4T_H \approx 0.159$
Li et al.	2019	[31]	$2T_P + T_B + 7T_H + T_S \approx T_B + 1.021$	$T_P + 3T_H \approx 0.510$	$T_P + 8T_H \approx 0.513$	$2T_P + 4T_H \approx 1.018$
Sharif et al.	2019	[32]	$T_B + 6T_H \approx T_B + 0.004$	$5T_H \approx 0.003$	$17T_H \approx 0.012$	$5T_H \approx 0.003$
Srinivas et al.	2018	[33]	$T_C + T_B + 11T_H \approx T_B + 0.516$	$T_C + 3T_H \approx 0.510$	$10T_H \approx 0.007$	$2T_C + 6T_H \approx 1.020$
Li et al.	2018	[3]	$2T_P + T_B + 5T_H \approx T_B + 0.019$	$T_P + 2T_H \approx 0.509$	$T_P + 6T_H \approx 0.512$	$2T_P + 4T_H \approx 1.019$
Wu et al.	2018	[34]	$5T_H \approx 0.003$	$6T_H \approx 0.004$	$15T_H \approx 0.010$	$5T_H \approx 0.003$
Amin et al.	2018	[35]	$4T_H \approx 0.003$	$8T_H \approx 0.006$	$15T_H \approx 0.010$	$6T_H \approx 0.004$
Wazid et al.	2017	[36]	$T_B + 4T_H \approx T_B + 0.003$	$5T_H + T_S \approx 0.004$	$11T_H + 2T_S \approx 0.009$	$7T_H + T_S \approx 0.005$
Park et al.	2016	[37]	$T_P + T_B + 6T_H \approx T_B + 0.512$	$T_P + 4T_H \approx 0.511$	$11T_H \approx 0.008$	$2T_P + 4T_H \approx 1.019$
Reddy et al.	2016	[23]	$2T_P + 2T_H \approx 1.017$	$T_P + 10T_H \approx 0.511$	$2T_P + 7T_H \approx 1.021$	$2T_P + 5T_H \approx 1.019$
Our scheme	–		$2T_P + T_B + 7T_H + T_I \approx T_B + 22.021$	$T_{be} + 3T_H \approx 9.354$	$5T_H + 2T_S \approx 0.005$	$T_{se} + 4T_H + T_S \approx 0.159$

Table 6 Communication and storage costs among relevant authentication schemes

Protocol	Year	Ref.	Communication cost I[a] (bits)			Communication cost II[b] (bits)			Storage cost (bits)		
			User	Gateway	Sensors	User	Gateway	Sensors	User	Gateway	Sensors
Wang et al.	2019	[30]	288	2560	128	3392	2656	1344	2528	1408	288
Li et al.	2019	[31]	416	832	128	640	928	480	800	416	288
Sharif et al.	2019	[32]	128	320	448	1216	1056	448	608	1280	320
Srinivas et al.	2018	[33]	320	320	128	928	768	3448	736	3328	288
Li et al.	2018	[3]	416	640	128	640	960	480	896	1312	288
Wu et al.	2018	[34]	288	864	128	864	1408	736	640	1408	416
Amin et al.	2018	[35]	288	768	128	864	1120	320	576	1440	288
Wazid et al.	2017	[36]	288	448	128	576	1184	608	864	2432	288
Park et al.	2016	[37]	288	768	128	736	1216	448	608	576	288
Reddy et al.	2016	[23]	160	928	128	928	1088	320	768	320	288
Our scheme	–		288	704	128	2688	1812	1344	928	1184	288

a) Communication cost I records the communication costs during the registration phase.
b) Communication cost II records the communication cost during the login and authentication phase.

time for biometric verification is 1.00 s [30], and this computation cost on the user side is acceptable. To finish the last step of the authentication phase, the user costs 9.354 ms, and this increase in time is too short to be noticeable (and thus acceptable).

From the communication and storage cost perspective as shown in Table 6, the communication costs in registration phase of the schemes in the table is close, except for Wang et al.'s scheme [30] where its gateway costs 2592 bits in registration phase. As for the communication costs in login and authentication phase, our cost on gateways is comparable to other schemes, and the cost on user sides and sensor nodes is slightly above average. The storage cost of our scheme is comparable to other schemes. Note that, Wang et al.'s scheme [30] and our scheme both exploit the imbalanced computational nature of the RSA cryptosystem to achieve forward secrecy with low computation cost on sensor nodes (0.159 ms). However, the communication and storage cost of our scheme is obviously smaller than their scheme, the main reason of this is that we combine ECC-based cryptosystem with RSA-based cryptosystem. The 160-bit ECC-based cryptosystem provides the same security level as that for 1024-bit RSA-based cryptosystem, so our

scheme cost less communication and storage workload.

In conclusion, compared with other schemes, the computation cost of our scheme on the user side is higher than most comparison schemes, and our scheme requires that the user terminal can support RSA cryptographic algorithm and ECC algorithm, simultaneously. As we analyzed above, this small defect is acceptable when the smart phone is getting popular. Besides, our scheme's communication costs on user side and sensor nodes are a little bit higher than some of the schemes. However, our scheme meets all eleven security requirements under the harshest adversary model; our computation cost on gateways is the smallest among all the comparison schemes; our computation cost on sensor nodes is comparable to those insecure schemes that use only symmetric cryptographic algorithms, and is superior to those that use asymmetric cryptographic algorithms. In brief, our scheme provides a good balance between performance and security, and is more suitable for the systems that have a mass of resource-constrained sensor nodes.

7 Conclusion

To ensure the sensitive data stored in sensor nodes cannot be accessed by any unauthorized entity, numerous user authentication schemes for industry 4.0 were proposed. However, most of them cannot balance the conflicts between security and efficiency, and the major challenge is to achieve forward secrecy in a resource-constrained sensor nodes environment. In this paper, we exploit the RSA cryptosystem's computational imbalance at the encryption side and the decryption side, and make full use of the computation and storage capability of the cloud center to design a cloud-aided efficient user authentication scheme with forward secrecy for industry 4.0. We formally prove its security under the random oracle model, and compare it with state-of-the-art relevant schemes. The results show the superiority of our scheme.

Acknowledgements This work was supported by the National Key Research and Development Plan of China (Grant No. 2018YFB0803605) and National Natural Science Foundation of China (Grant No. 61802006).

References

1 Kemmerer S. Manufacturing Interoperability Program: A Synopsis. National Institue of Standards and Technology, Technical Report, 2009
2 Garg S, Kaur K, Kaddoum G, et al. Toward secure and provable authentication for internet of things: realizing industry 4.0. IEEE Internet Things J, 2020, 7: 4598–4606
3 Li X, Niu J, Bhuiyan M Z A, et al. A robust ECC-based provable secure authentication protocol with privacy preserving for industrial Internet of Things. IEEE Trans Ind Inf, 2018, 14: 3599–3609
4 Kumari S, Karuppiah M, Das A K, et al. A secure authentication scheme based on elliptic curve cryptography for IoT and cloud servers. J Supercomput, 2018, 74: 6428–6453
5 Lin C, He D, Kumar N, et al. Security and privacy for the Internet of drones: challenges and solutions. IEEE Commun Mag, 2018, 56: 64–69
6 Wang D, Li W, Wang P. Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks. IEEE Trans Ind Inf, 2018, 14: 4081–4092
7 Wang C, Wang D, Tu Y, et al. Understanding node capture attacks in user authentication schemes for wireless sensor networks. IEEE Trans Depend Secure Comput, 2020. doi: 10.1109/TDSC.2020.2974220
8 Das A K, Wazid M, Kumar N, et al. Design of secure and lightweight authentication protocol for wearable devices environment. IEEE J Biomed Health Inform, 2018, 22: 1310–1322
9 Bresson E, Chevassut O, Pointcheval D. Security proofs for an efficient password-based key exchange. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, 2003. 41–50
10 Wang D, Wang P. Two birds with one stone: two-factor authentication with security beyond conventional bound. IEEE Trans Depend Secure Comput, 2018, 15: 708–722
11 He D, Zeadally S, Kumar N, et al. Efficient and anonymous mobile user authentication protocol using self-certified public key cryptography for multi-server architectures. IEEE Trans Inform Forensic Secur, 2016, 11: 2052–2064
12 Meng W, Li W, Jiang L, et al. Socialauth: designing touch behavioral smartphone user authentication based on social networking applications. In: Proceedings of IFIP International Conference on ICT Systems Security and Privacy Protection, 2019. 562: 180–193
13 Meng W, Wang Y, Wong D S, et al. TouchWB: touch behavioral user authentication based on web browsing on smartphones. J Network Comput Appl, 2018, 117: 1–9
14 Wu T D. The secure remote password protocol. In: Proceedings of Internet Society Symposium on Network and Distributed System Security, 1998. 98: 97–111
15 Das M L. Two-factor user authentication in wireless sensor networks. IEEE Trans Wirel Commun, 2009, 8: 1086–1090
16 Fan R, He D, Pan X, et al. An efficient and DoS-resistant user authentication scheme for two-tiered wireless sensor networks. J Zhejiang Univ Sci C, 2011, 12: 550–560
17 Wang D, Wang P. On the anonymity of two-factor authentication schemes for wireless sensor networks: attacks, principle and solutions. Comput Netw, 2014, 73: 41–57
18 Das A K, Sharma P, Chatterjee S, et al. A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. J Network Comput Appl, 2012, 35: 1646–1656
19 Xue K, Ma C, Hong P, et al. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. J Network Comput Appl, 2013, 36: 316–323

20 Ma C G, Wang D, Zhao S D. Security flaws in two improved remote user authentication schemes using smart cards. Int J Commun Syst, 2014, 27: 2215–2227
21 Choi Y, Lee D, Kim J, et al. Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors, 2014, 14: 10081–10106
22 He D, Kumar N, Chilamkurti N. A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. Inf Sci, 2015, 321: 263–277
23 Reddy A G, Das A K, Yoon E J, et al. A secure anonymous authentication protocol for mobile services on elliptic curve cryptography. IEEE Access, 2016, 4: 4394–4407
24 Jiang Q, Zeadally S, Ma J, et al. Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks. IEEE Access, 2017, 5: 3376–3392
25 Wang D, He D, Wang P, et al. Anonymous two-factor authentication in distributed systems: certain goals are beyond attainment. IEEE Trans Depend Secure Comput, 2015, 12: 428–442
26 Dolev D, Yao A. On the security of public key protocols. IEEE Trans Inform Theor, 1983, 29: 198–208
27 Wang D, Wang P. On the implications of Zipf's law in passwords. In: Proceedings of European Symposium on Research in Computer Security, 2016. 111–131
28 Li W, Wang D, Wang P. Insider attacks against multi-factor authentication protocols for wireless sensor networks (in Chinese). J Softw, 2019, 30: 2375–2391
29 Wang D, Cheng H, Wang P, et al. Zipf's law in passwords. IEEE Trans Inform Forensic Secur, 2017, 12: 2776–2791
30 Wang D, Wang P, Wang C. Efficient multi-factor user authentication protocol with forward secrecy for real-time data access in WSNS. ACM Trans Cyber-Phys Syst, 2020, 4: 30
31 Li X, Peng J, Obaidat M S, et al. A secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems. IEEE Syst J, 2020, 14: 39–50
32 Sharif A O, Arshad H, Nikooghadam M, et al. Three party secure data transmission in IoT networks through design of a lightweight authenticated key agreement scheme. Future Gener Comput Syst, 2019, 100: 882–892
33 Srinivas J, Das A K, Wazid M, et al. Anonymous lightweight chaotic map-based authenticated key agreement protocol for industrial Internet of Things. IEEE Trans Depend Secure Comput, 2020, 17: 1133–1146
34 Wu F, Li X, Sangaiah A K, et al. A lightweight and robust two-factor authentication scheme for personalized healthcare systems using wireless medical sensor networks. Future Gener Comput Syst, 2018, 82: 727–737
35 Amin R, Islam S H, Biswas G P, et al. A robust and anonymous patient monitoring system using wireless medical sensor networks. Future Gener Comput Syst, 2018, 80: 483–495
36 Wazid M, Das A K, Odelu V, et al. Secure remote user authenticated key establishment protocol for smart home environment. IEEE Trans Depend Secure Comput, 2020, 17: 391–406
37 Park Y H, Park Y H. Three-factor user authentication and key agreement using elliptic curve cryptosystem in wireless sensor networks. Sensors, 2016, 16: 2123–2140
38 Srinivas J, Mukhopadhyay S, Mishra D. Secure and efficient user authentication scheme for multi-gateway wireless sensor networks. Ad Hoc Netw, 2017, 54: 147–169

Supporting information Appendix A. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

Appendix A Formal security analysis of our scheme

In the proof below, we prove Theorem 1 via a sequence of games starting at a real attack game G_0 and ending up with G_6 where the advantage of the adversary to attack the protocol is zero.

Game G_0: It is a real attack, and thus we have

$$\mathrm{Adv}_{\mathcal{P}}^{\mathrm{ake}}(A) = 2\Pr[\mathrm{Succ}_0] - 1, \tag{A1}$$

where Succ_n is the event that $\mathcal{A}$ in Game G_n correctly guesses b in Test$(\cdot)$-query.

Game G_1: This game simulates several hash functions, $\mathcal{H}_i : \{0,1\}^* \to \{0,1\}^l$ $(i = 1, 2)$, and $\mathcal{H}_i'$. Furthermore, all the queries are simulated as the real protocol, three lists need to be maintained: $\Lambda_{\mathcal{H}}$ which records the replies to hash queries; $\Lambda_{\mathcal{M}}$ which stores the transcripts in those games; $\Lambda_{\mathcal{A}}$ which stores hash-queries asked by the adversary $\mathcal{A}$; Λ_{ε} which stores encryption-queries asked by the adversary $\mathcal{A}$; $\Lambda_{\mathcal{D}}$ which stores decryption-queries asked by the adversary $\mathcal{A}$. Then we have

$$|\Pr[\mathrm{Succ}_1] - \Pr[\mathrm{Succ}_0]| = 0. \tag{A2}$$

Game G_2: In this game, we remove collisions.

- The probability of collisions caused by the hash function is $\frac{q_h^2}{2^{l+1}}$.
- The probability for collisions of k_x, k_g and k_j is $\frac{(q_s+q_e)^2}{2^{l+1}}$.
- The probability for collisions of k_i is at most $\frac{(q_s+q_e)^2}{2p}$.

Thus, we have

$$|\Pr[\mathrm{Succ}_2] - \Pr[\mathrm{Succ}_1]| \leqslant \frac{(q_s+q_e)^2}{2p} + \frac{q_h^2 + (q_s+q_e)^2}{2^{l+1}}. \tag{A3}$$

Game G_3: This game considers such an attack where $\mathcal{A}$ luckily guesses messages correctly without querying hash oracle.

- For MSG_1, the simulator checks whether $(\mathrm{ID}_i||\mathrm{SID}_j||*, *)$, $(*||\mathrm{ID}_i||*||*||\mathrm{SID}_j||*, M_2)$, $(\mathrm{ID}_i||*, *) \in \Lambda_{\mathcal{A}}$, and the probability of these items in $\Lambda_{\mathcal{A}}$ is $\frac{(q_s+2q_h)^2}{2^l}$.
- For MSG_2, the simulator checks whether $(*||*||\mathrm{GID}_k||\ \mathrm{SID}_j||*, M_4) \in \Lambda_{\mathcal{A}}$, and the probability is $\frac{q_s^2}{2^l}$.
- For MSG_3, the simulator checks whether $(*||*||*||\ \mathrm{SID}_j||\mathrm{GID}_k||*, M_6) \in \Lambda_{\mathcal{A}}$, and the probability is $\frac{q_s^2}{2^l}$.

- For MSG_4, the simulator checks whether $(*||*||*||\ \mathrm{SID}_j, *)$, $(*||\mathrm{SID}_j||\mathrm{GID}_k||*||*||M_7, M_8)$ and $(*||*||M_7, M_9) \in \Lambda_{\mathcal{A}}$, and the probability is $\frac{(2q_s+q_h)^2}{2^l}$.
- For MSG_5, the simulator checks whether $(*||*||\mathrm{GID}_k\ ||\mathrm{SID}_j||*||M_7||M_9, M_{10}) \in \Lambda_{\mathcal{A}}$, and the probability is $\frac{q_s^2}{2^l}$.
- For MSG_6, the simulator checks whether $(*||*||*||\mathrm{ID}_i\ ||\mathrm{SID}_j||*||*||M_7||M_9, M_{11}) \in \Lambda_{\mathcal{A}}$, the probability is $\frac{q_s^2}{2^l}$.

Thus, we have

$$|\Pr[\mathrm{Succ}_4] - \Pr[\mathrm{Succ}_3]| \leqslant \frac{(q_s+2q_h)^2 + (2q_s+q_h)^2 + 4q_s^2}{2^l}. \tag{A4}$$

Game G_4: In this game, $\mathcal{A}$ computes d_i correctly. Games G_4 and G_3 are indistinguishable unless the event where $\mathcal{A}$ computes d_i via hash oracle with $(\mathrm{ID}_i||\mathrm{RPW}_i)$ or $(\mathrm{ID}_i||y||x_i)$ happens, and we denote such an event as AskPara, then,

$$|\Pr[\mathrm{Succ}_4] - \Pr[\mathrm{Succ}_3]| \leqslant \Pr[\mathrm{AskPara}_4]. \tag{A5}$$

$\mathcal{A}$ only has two ways to compute b_i: asking $\mathrm{Corrupt}(u, a)$. It should note that $\mathcal{A}$ can only ask one kind of $\mathrm{Corrupt}(u, a)$ query, and we denote the possibility of $\mathcal{A}$ computing b_i by using $\mathrm{Corrupt}(u, a)$ successfully as $\Pr[\mathrm{AskPara}_4\mathrm{withCorr}_a]$, where $a = 1, 2, 3$. Then we have

$$\begin{aligned}
|\Pr[\mathrm{AskPara}_5\mathrm{withCorr}_1]| &\leqslant \frac{q_s}{2^l},\\
|\Pr[\mathrm{AskPara}_5\mathrm{withCorr}_2]| &\leqslant \frac{q_s}{2^l} + q_s \cdot P_{\mathrm{fasle}},\\
|\Pr[\mathrm{AskPara}_5\mathrm{withCorr}_3]| &\leqslant C' q_{\mathrm{send}}^{s'},
\end{aligned}$$

where P_{false} is the probability of two persons who have similar fng_i. C' and s' are Zipf parameters [29].

According to the above equations, we have

$$|\Pr[\mathrm{AskPara}_4]| \leqslant \max\{|\Pr[\mathrm{AskPara}_4\mathrm{withCorr_a}]|\} = C' q_{\mathrm{send}}^{s'}. \tag{A6}$$

Game G_5: This game applies the elliptic curve gap Diffie-Hellman problem, and makes M_1 and SK be independent for d_i and k_i. In this game, the adversary tries to compute the correct k_j and K_2 with the history transcript, i.e., using the information from Execute$(\cdot)$ queries.

$$|\Pr[\mathrm{Succ}_5] - \Pr[\mathrm{Succ}_4]| \leqslant q_h(\mathrm{Adv}_p^{\mathrm{ECDH}}(t') + \mathrm{Adv}_p^{\mathrm{CDH}}(t')), \tag{A7}$$

where $t' \leqslant t + (q_s + q_e + 1) \cdot T_m$

Game G_6: This game considers forward secrecy. Thus $\mathcal{A}$ can query $\mathrm{Corrupt}(S)$ or $\mathrm{Corrupt}(G)$. After having asked one of these two queries $\mathcal{A}$ can only ask Execute$(\cdot)$-query. Thus,

$$|\Pr[\mathrm{Succ}_6] - \Pr[\mathrm{Succ}_5]| \leqslant 2q_h(q_s+q_e)^2 \cdot (\mathrm{Adv}_{\mathcal{A}}^{\mathrm{ECDH}}(t') + \mathrm{Adv}_p^{\mathrm{CDH}}(t')). \tag{A8}$$

Till now, there is no advantage for $\mathcal{A}$ to guess the session key.