A Lightweight, Secure Big data-based Authentication and Key-agreement Scheme for IoT with Revocability

Behnam Zahednejad[1]

Institute of Artificial Intelligence and Blockchain, Guangzhou University , Guangzhou, China; bzahednezhad@gmail.com

Abstract. With the rapid development of Internet of Things (IoT), designing a secure two-facor authentication scheme for these network is increasingly demanding. Recently, historical bigdata has gained interest as a novel authentication factor in this area. In this paper, we focus on a recent authentication scheme using bigdata (Liu et al.'s scheme) which claims to provide additional security properties such as Perfect Forward Secrecy (PFS), Key Compromise Impersonation (KCI) resilience and Server Compromise Impersonation (SCI) resilience. However, assuming a real strong attacker, rather than a weak one. we show that their scheme not only fails to provide KCI and SCI, but also doesn't provide real two-factor security, revocability and suffers inside attack. Then we propose our novel scheme which can indeed provide two-factor security, PFS , KCI and inside attack resilience and revocability of the client. Further, our performance analysis shows that our scheme has reduced modular exponentiation operation and multiplication for both client and server compared to Liu et al.'s scheme which reduces the execution time by one third i.e. 6 ms and 30 ms (0.3 ms and 4 ms) for IoT device (server) for security levels of $\lambda = 128, \lambda = 256$ respectively.

Keywords: Internet of Things (IoT) · historical bigdata · Key Compromise Impersonation (KCI) resilience · Perfect Forward Secrecy (PFS) · Inside attack · revocability

1 Introduction

Internet of Things (IoT) has enabled a wide range of objects in our world to access network connectivity, send and receive data. These IoT devices have limited power, storage, and processing capabilities. In addition, they are often deployed in public and hostile environment, which expose them to a wide range of attacks such as physical and cloning attacks. To provide security in these networks, cryptographic solutions are among the key technologies, which can guarantee authentication and key agreement between IoT devices and the server.

To this end, single factor authentication schemes such as password-based or secret-key based schemes, in which a shared secret is the only authentication factor, are no longer sufficient for addressing the security requirements. Given the advances made in physical or side-channel attacks, the adversary can obtain IoT device's secret, and thus compromise the entire system.

Two factor authentication schemes have been proposed to resolve these threats and add an extra defense layer in order to provide a more resilient way of authenticating IoT devices.

Recently, big data generated by IoT devices at a great velocity has been adopted as an authentication factor by researchers [1,2]. In this paper, we focus on Liu's scheme [2] which claims to provide various security goals such as Key Compromise Impersonation (KCI) resilience , Perfect Forward Secrecy (PFS), Server Compromise Impersonation (SCI) resilience in addition to standard goals. Assuming a real attacker who can obtain one security factor, we show that their scheme fails to provide real two-factor security. Further, the adversary can mount KCI , SCI attack and inside attack.

1.1 Related works

So far, the existing two-factor authentication schemes have used either the following factors or a combination of them:

1. What you know (password): Password is the most conventional method of providing security and authentication. However, it is prone to many attacks such as loss of password, eavesdropping,password guessing attacks , forgetting passwords, etc [3].
2. What you are (biometric): In this method, user's identity is recognized using their fingerprint, facial features, hand shape, iris structure, voice etc [4,5]. The main challenge to widespread implementation of this method is its cost. Also, they are unrecoverable once compromised. For instance, at GeekPwn 2019 conference in Shanghai, it was shown how to create and use a photograph of a user's fingerprint to unlock their smartphone in less than 20 minutes[6].
3. What you have (smart card, PUF): Given the challenges of the above factors, researchers adopted additional factors to enhance security. For instance, Jiang et al. [7] offered a novel user authentication protocol.However, Wen et al. [8] showed that his scheme suffers spoofing and replay attacks. Then he proposed an improved scheme which resists against such attacks. Later, Gope and Hwang [9] illustrated serious flaws of Wen et al.'s scheme, including offline password guessing attack and user forgery attack. Although these works are among lightweight and symmetric schemes, in all these schemes the key information need to be stored in the smart card, which costs a large memory overhead. PUF-based authentication schemes provide a novel solution with much less memory and performance overhead. In PUF-based schemes, the challenge-response pairs aren't stored directly. Aman et al. [10] designed a PUF-based authentication scheme to encrypt the data using the response of the PUF. Similarly, Chatterjee et al. [11] proposed a PUF-based scheme to construct the session key using the response of the PUF. However, these two PUF-based protocols fail to provide anonymity. This issue was resolved in

Feikken et al. [12] and Gope and Sikdar [13] schemes wherein the identity of the user is no longer transmitted in plaintext, rather a pseudo-identity is transferred along with the challenge-response pair. Although anonymity is resolved in this scheme, desynchronization attacks is still a problem. This issue is resolved in by Jiang et al. ‘s scheme [14] with the expense of an increased overhead due to the usage of asymmetric cryptography. Besides security, another shortage of existing PUF-based scheme is the ignorance of noisy factors in PUFs. Some schemes [15] introduced fuzzy extractors or reverse fuzzy extractors to address this noise issue.

4. Fourth factor: Given the special hardware requirement of the PUF or smart card based methods, and also in settings without human involvement, a fourth factor is needed as a general-purpose method. ”Whom you know” [16] and ”where you are” [17] methods have been suggested to fill this gap. In a parallel effort, historical big-data stored in the server over such a long time, has been suggested as a new factor (”what have been discussed”) . This method was pioneered by Chan et al. [1] who proposed a novel bigdata-based unilateral two-factor authentication scheme. The two factors include the shared long-term key and all available historical data and their corresponding tags, which are stored as data-tag tuples. They proved the security of their scheme in a bounded retrieval model where the adversary can only have access to part of the data-tag tuples. Following the same method, Liu et al. [2] introduced an enhanced authentication scheme which achieves more security properties including mutual authentication, forward secrecy, key compromise impersonation resilience and server compromise impersonation resilience.

1.2 Our contribution

In this paper, we first put in question the main security assumption of the recent bigdata-based schemes(Chan et al. [1] and Liu et al. [2]). They proved the security of their scheme assuming a weak type of adversary called bounded retrieval model who can only access a small fraction of the data-tag tuples (d_i, t_i) stored in the server, after compromising the server. This type of respectful adversary is a weak assumption and doesn’t exist in reality. Once the server is compromised, the attacker doesn’t differentiate between different items and steals the whole parameters and data-base. Then we focus on Liu et al.’s scheme and show that it can’t achieve its claimed security properties (including two-factor authentication, tag secrecy and KCI and SCI[1] resilience) in a real attacker setting. In addition, in real IoT scenarios, the server is usually in contact with different clients with different identities. However, in Liu et al.’s scheme, the client is anonymous which makes their scheme non-revocable and also prone to inside attack.

[1] This attack isn’t a standard attack against security of authentication protocols. Under a real attacker assumption, if the server is compromised, the attacker can obviously impersonate it. Hence, we neglect considering this attack in our analysis.

Secondly, we propose our improved big data-based scheme which satisfies truly two factor authentication under a real and strong attacker model. In addition, it achieves mutual authentication, perfect forward secrecy, key compromise impersonation resilience, resistance to inside attack, revocablity and unlinkability of the client. Thirdly, we prove the achievement of the mentioned security properties of our proposed scheme informally and formally using Real-Or-Random (ROR) model and Bellare and Rogaway model. Finally, we use Raspberry Pi 3 Model B+ as the IoT device and a PC as the server to implement our proposed scheme and compare its performance with previous schemes. The results indicate that the time complexity of our scheme is reduced by one third compared to Liu et al.

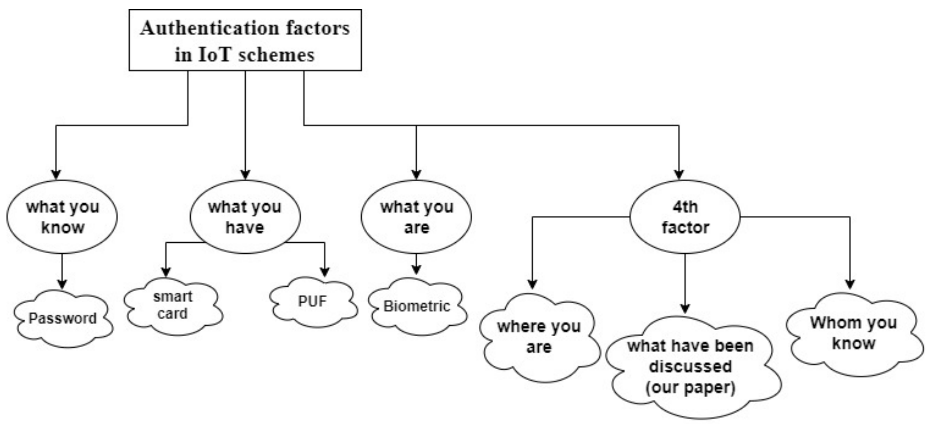

Fig. 1: Types of authentication factors

2 Preliminaries

For the integrity of the paper, we present the relevant preliminaries , threat model and security requirements of authentication protocols. In addition, a summary of notations is shown in Table 1.

2.1 Signer Efficient Multiple-time Elliptic Curve Signature (SEMECS)

Digital signatures are basic primitives which provide message authentication and non-reputation in security networks. However, typical digital signature schemes can not be directly used in IoT resource-constrained devices due to large private key and signature sizes.. Recently, researchers tried to design ultra-lightweight signature scheme for resource-constrained devices. In this paper, we use Yavuz et al.'s scheme [18] (Signer Efficient Multiple-time Elliptic Curve Signature (SEMECS)) as state of the art ultra lightweight signature scheme. This scheme falls

Table 1: Notations used through the paper.

Notation	Description	Notation	Description
s	Server	ssk_i	Secret signing key of party i
c	Client	spk_i	Public signing key of party i
$\mathbb{D}$	Server's data items	mk	shared secret key
ID_c	Identity of client c	SK	shared session key
TID_c	Pseudo-Identity of client c	$H_0(.), H_1(.), H(.)$	Distinctive Hash functions $\{0,1\}^* \rightarrow \mathbb{Z}_q$
$\mathbb{I}_i$	Subset of data-items indices chosen by i	r_i	i-th random number
$\oplus$	Bitwise Xor operation	$\|\|$	String concatenation operation
L	Size of the server's big data	z	Size of the sub-set $\mathbb{I}_i$

into the category of k-time signature schemes wherein after K signatures, the signing key pair must be re-generated. Algorithm 1 describes the key generation, signature and verification of SEMECS:

Algorithm 1: Signer Efficient Multiple-time Elliptic Curve Signature (SEMECS) Scheme

$(sk_0, PK) \leftarrow$ SEMECS.Kg $(1^K, K)$

1. Generate large primes q and $p > q$ such that $q|(p-1)$
2. Select a generator α of the subgroup $\mathbb{G}$ of order q in $\mathbb{Z}_p^*$
3. Set a private/public key pair $(y \leftarrow \mathbb{Z}_q^*, Y = \alpha^y \bmod p)$ and system parameter $I \leftarrow (q, p, \alpha)$ as the output.
4. **for** j=0,...,K-1 **do**:
5. $r_j \leftarrow H_0(y||j)$, $R_j \leftarrow \alpha^{r_j} \bmod p$, $z_j \leftarrow H_1(y||j)$
   $\gamma_j \leftarrow z_j \oplus H_0(R_j)$, $\beta_j \leftarrow H_1(R_j)$
6. **Return** $sk_0 \leftarrow y$ and $PK = (Y, \alpha, , ((\gamma_0, \beta_0), ..., (\gamma_{K-1}, \beta_{K-1})), K)$

$\sigma \leftarrow SEMECS.Sig(sk_j = (y, j), M_j)$

1. **if** $|M_j| < |q|$ then set $(\bar{M}_j = M_j, \hat{M}_j = 0)$
   else split M_j into two as $(\bar{M}_j||\hat{M}_j)$ such that $\bar{M}_j = |q|$.
2. $r_j \leftarrow H_0(y||j), z_j \leftarrow H_1(y||j), c_j \leftarrow \bar{M}_j \oplus z_j$
3. $e_j \leftarrow H_0(c_j||\bar{M}_j)$ $s_j \leftarrow r_j - e_j.y \bmod q$
4. **if** $j > K-1$ **then return** $\perp$
5. **else return** $\alpha_j \leftarrow (s_j, c_j)$ The value of $\alpha_j, \hat{M}_j$ is given to the receiver.

$b \leftarrow SEMECS.Ver(PK, \hat{M}_j, \alpha_j)$ If $|c_j| > |q|$ or $|j| \geq |q|$, return 0. Otherwise:

1. $R_j^{'} \leftarrow Y^{H_0(c_j||\hat{M}_j)}.\alpha^{s_j} modp.$
2. If $\beta_j \neq H_1(R_j^{'})$ then return $b = 0$.
3. else return $b = 1$ and recover M_j as follows:

4. $\bar{M}_j \leftarrow \gamma_j \oplus H_0(R_j^{'}) \oplus c_j$
5. If $\hat{M}_j = 0$ then set $M_j = \bar{M}_j$.

The security of the SEMECS scheme is based on the DLP problem:
Definition 1: Discrete Logarithm Problem (DLP). Let p be a prime value and a, b be non-zero integers (mod p). The problem of finding x such that $a^x = b$ (mod p) within polynomial time is a hard task.

2.2 Threat model & security factors

Previous big-data based schemes (Chan et al. & Liu et al.) assumed a bounded retrieval model wherein the attacker could only access a small portion of the data items stored in the server's database after compromising the server. In this paper, we adopt a stronger adversary who can obtain the whole data items after compromising the server. In addition, we assume the following capabilities for the adversary.

1. The adversary $\mathcal{A}$ can obtain the whole secrets stored in the IoT device using side-channel techniques.
2. $\mathcal{A}$ can block, intercept, modify, delete, and resend any message transmitted through the public channel.
3. $\mathcal{A}$ can only obtain one of the two security factors, but not both of them simultaneously. The main two security factors used in our scheme include the long-term secrets of the server and client respectively.
4. In case of carrying out perfect forward secrecy attack, we assume that $\mathcal{A}$ can obtain the long-term secrets of both the client and server.

In addition, we use the following factors as the security basis of our proposed scheme:

1. The secrets stored in the server side i.e. $\mathcal{S}_s = \{mk, ssk_s, \mathbb{D}\}$
2. The secrets stored in the client side i.e. $\mathcal{S}_c = \{mk, ssk_c\}$

2.3 Security Requirements in IoT authentication schemes

1. **Mutual authentication**: The communicating parties should authenticate and verify each other's legitimacy before exchanging secret messages. This is the most basic requirement which prevents adversaries from impersonating legitimate parties.
2. **Key Agreement with Secrecy**: As IoT networks are deployed in various applications including healthcare, industry and military purposes, sensitive data such as user's identities, secrets keys and confidential commands need to be private and shared only between legitimate parties. Therefore, after completing the authentication, the communicating parties should establish a shared session key to safeguard their secret information from adversary.

3. **Two-factor security**: In order to increase the security level of authentication schemes, two-factor security requires that even if one of the security factors is leaked, security properties shouldn't be violated.
4. **Revocability**: According to this requirement, if one of the devices is lost or stolen, the server should be able to revoke its legitimacy without a great change for the whole network.
5. **Anonymity and unlinkability**: In most scenarios, not only the real identity of the device should be hidden (anonymity), but also the adversary should not be able to find any link or relation between the pseudo-identity of the suspected device in different sessions. Otherwise, the suspected device can be easily traced. This requirement is known as unlinkability.
6. **(Perfect) forward secrecy**: Forward secrecy is a well-known requirement in authentication protocol which ensures that if long-term parameters of one party (such as long-terms keys, the values stored in devices, etc.) is revealed in one session, the session keys of previous sessions shouldn't be disclosed to adversary.
   Perfect forward secrecy also requires the secrecy of previous session keys assuming the disclosure of both parties long-term parameters. This property can be achieved easily by using public-key operations such as diff-hellman. However using public-key operations in tiny IoT devices is not feasible due to its high computation overhead.
7. **Key Compromise Impersonation (KCI) resilience**: KCI resilience requires that the compromise of one party shouldn't let the attacker impersonate other entities to that party. In another word, a key agreement protocol is KCI-resilient if compromise of the long-term parameters of a specific principal does not give any chance to the attacker to construct a session key with that principal through impersonation as a different principal.
8. **Resistance to privileged inside attack**: In IoT networks, the server is usually in contact with billions of tiny IoT devices, wherein the chance of their compromise is quite high. According to this requirement, not a single hostile device should be able to impersonate other devices.
9. **Resistance to well-known attacks**: Every security protocol in IoT network should resist against well-known attacks such as Man-In-The-Middle (MITM), replay , DoS , etc. These attacks target the basic requirements of authentication protocols.

3 Review of Liu et al.'s authentication scheme

3.1 Initialization phase

In this phase, the security parameter λ e.g. ($\lambda = 128$ or $\lambda = 256$) is chosen by the server **s**. Also the public parameters are initialized as follows: a group $\mathbb{G}$ of prime order q, a generator of g of $\mathbb{G}$, a cryptographic hash function $H : \{0,1\}^* \rightarrow \mathrm{Z}_q$ and two Pseudo-Random Functions (PRFs) $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \rightarrow \mathrm{Z}_q, E : \{0,1\}^\lambda \times \{0,1\}^\lambda \rightarrow \{0, 1\}^\lambda$. In addition, the server **s** produces the public/private key pair $(pk = g^{sk}, sk)$ wherein $sk \in \mathbb{Z}_q$. It also generates $sk_1 = \{mk\}$ where $mk \in \{0,1\}^{\lambda}$ as a long-term shared key between the IoT device and the server and $sk_2 = \{K, K^{'}\}$ for tag generation and data processing where $K \in \mathbb{Z}_q$ and $K^{'} \in \{0,1\}^{\lambda}$. The server holds a dataset $\mathbb{D}$ with L data items $d_i (0 \leq d_i \leq L)$. A tag $t_i = K.H(d_i) + F_{K'}(i)$ is generated for each data item d_i by the server. The dataset $\mathbb{D}^*$ is defined as a container for all data item and tag tuples (d_i, t_i) .

The index parameter z is also chosen by the server.

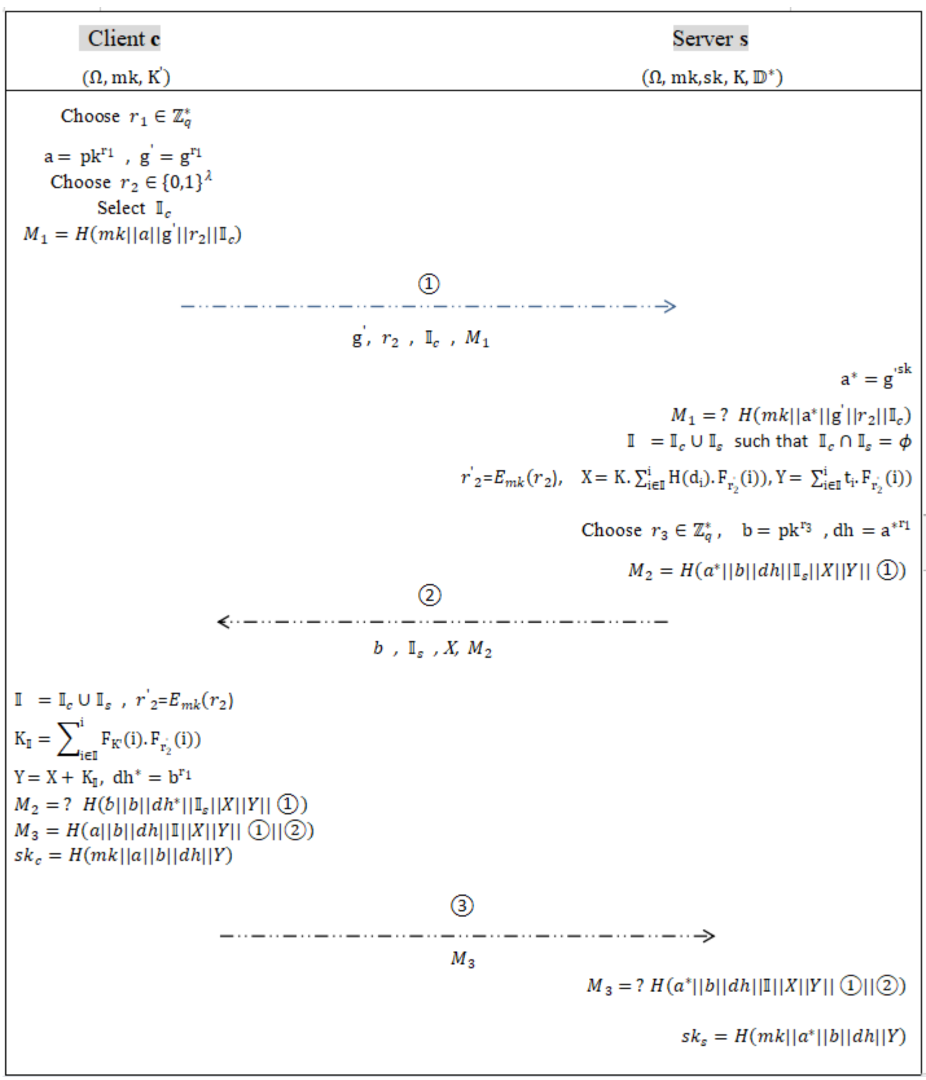

Fig. 2: Liu's authentication phase

3.2 Authentication phase

The authentication procedure between the IoT device **c** and the server **s** is summarized as follows:

1. The IoT device **c** chooses random number $r_1 \in \mathbb{Z}_q^*$ to compute $a = pk^{r_1}$ and $g^{'} = g^{r_1}$. Then it chooses $r_2 \in \{0,1\}^{\lambda}$ and a random subset $\mathbb{I}_c$ of z distinct indices for the tuples in $\mathbb{D}^*$. Then it sends $g^{'}, r_2, \mathbb{I}_c$ and $M_1 = H(mk||a||g^{'}||r_2||\mathbb{I}_c)$ to the server **s**.
2. After receiving the above message, the server **s** calculates $a^* = (g^{'})^{sk}$ and checks if the relation $M_1 = H(mk||a^*||g^{'}||r_2||\mathbb{I}_c)$ holds or not. If it holds, the server **s** chooses a subset $\mathbb{I}_s$ of z distinct indices disjoint from $\mathbb{I}_c(\mathbb{I}_s \cap \mathbb{I}_c = \emptyset)$. Then the values of $r_2^{'} = E_{mk}(r_2), X = K.\sum_{i \in \mathbb{I}}^{i}(H_(d_i).F_{r_2^{'}}(i))$ and $Y = \sum_{i \in \mathbb{I}}^{i}(t_i.F_{r_2^{'}}(i))$ are computed where $\mathbb{I} = \mathbb{I}_c \bigcup \mathbb{I}_s$. Here, the values of X, Y are computed in the finite field $\mathbb{Z}_q$. In addition, it chooses the random value r_3 to compute $b = pk^{r_3}, dh = a^{*r_3}$. Finally, the values of $b, \mathbb{I}_s, X$ and $M_2 = H(a^*||b||dh||\mathbb{I}_s||X||Y||\textcircled{1})$ are sent to the IoT device **c**. Here, $\textcircled{1}$ represents the messages transmitted in the first round, i.e $g^{'}, r_2, \mathbb{I}_c, M_1$.
3. After the IoT device **c** receives the message, it computes $r_2^{'} = E_{mk}(r_2)$ and $k_{\mathbb{I}} = \sum_{i \in \mathbb{I}}^{i}(F_{K'}(i).F_{r_2^{'}}(i))$, where $\mathbb{I} = \mathbb{I}_c \bigcup \mathbb{I}_s$. Next, it computes $Y = X + k_{\mathbb{I}}$ and $dh^* = b^{r_1}$ verifies if $M_2 = H(a||b||dh^*||\mathbb{I}_s||X||Y||\textcircled{1})$ holds. If this relation holds, it computes $M_3 = H(a||b||dh^*||\mathbb{I}||Y||\textcircled{1}||\textcircled{2})$ and transmits it to the server **s**. Finally, the IoT device **c** calculates the session key and session identifier as $SK_c = H(mk||a||b||dh^*||Y)$ and $SID_c = H(\textcircled{1}||\textcircled{2})$ respectively.
4. Upon receiving the message, the server **s** checks if the relation $M_3 = H(a^*||b||dh||\mathbb{I}||Y||\textcircled{1}||\textcircled{2})$ holds or not. If it holds, its session key and session identifier are calculated as $SK_s = H(mk||a^*||b||dh||Y)$ and $SID_s = H(\textcircled{1}||\textcircled{2})$ respectively.

Figure 2 represents a schematic of Liu et al.'s authentication phase.

3.3 On the Flaws of Liu et al.'s authentication scheme

Not truly two factor security Two-factor security requires that the security properties shouldn't be violated in case of compromising one factor. In the Liu et al.'s scheme, the two security factors are:

- The long-term private key mk shared between the client and the server.
- The server's datasets which contains a large number of data items denoted as d_i along with their corresponding tags t_i which are stored as tuples (d_i, t_i) in the server's database , $\mathbb{D}^*$.

Liu et al. clearly claim that the adversary can't forge the tags after compromising the server. However, they assume a weak model of adversary called bounded retrieval model who can only obtain a small fraction of the data and tag tuples (d_i, t_i), when it compromises the server. In this paper, we focus on a stronger type of adversary, who not only steals the whole datasets, but also obtains the other factor i.e. the long-term shared private key mk, after compromising the server . In other words, the security of the Liu et al.'s scheme is based on a single factor i.e. $\mathcal{S}_s = \{mk, sk, K, \mathbb{D}^* = (d_i, t_i)\}$. In this strong model, other security requirements such as KCI no longer holds.

Being prone to KCI attack According to the KCI definition [19], even if the long-term secrets of one party are revealed, the attacker shouldn't have any chance to impersonate other parties to the corrupted one. Despite Liu et al.'s claim, their scheme can't be KCI resilient. Once the server **s** is corrupted, the attacker can impersonate the client **c**. To this end, the attacker who is given long-term parameters of the server, $\mathcal{S}_s = \{mk, sk, K, \mathbb{D}^* = (d_i, t_i)\}$, chooses a random number $r_1 \in \mathbb{Z}_q^*$ to compute $a = pk^{r_1}$ and $g^{'} = g^{r_1}$. Then it chooses $r_2 \in \{0,1\}^{\lambda}$ and a random subset $\mathbb{I}_c$ of z distinct indices for the tuples in $\mathbb{D}^*$. Then it sends $g^{'}, r_2, \mathbb{I}_c$ and $M_1 = H(mk||a||g^{'}||r_2||\mathbb{I}_c)$ to the server **s**. After receiving the server's response, i.e $b, \mathbb{I}_s, X, M_2$, it computes $r_2^{'} = E_{mk}(r_2), dh^* = b^{r_1}$ In addition, using server's long-term parameters, it computes $Y = \sum_{i\in\mathbb{I}}^{i}(t_i.F_{r_2'}(i))$. Then, it computes $M_3 = H(a||b||dh^*||\mathbb{I}||Y||\textcircled{1}||\textcircled{2})$ and transmits it to the server **s**. Finally, it calculates the session key and session identifier as $SK_c = H(mk||a||b||dh^*||Y)$ and $SID_c = H(\textcircled{1}||\textcircled{2})$ respectively and makes further connection with the server using this session key. After receiving the attacker's message, the server **s** accepts the attacker as a legitimate client **c** as the relation $M_3 = H(a^*||b||dh||\mathbb{I}||Y||\textcircled{1}||\textcircled{2})$ holds.

Inefficient data items authentication In order to authenticate data items $d_i(0 \leq i \leq L)$ for the IoT device in both Chan et al. and Liu et al.'s scheme, each data item (d_i) has a corresponding tag (t_i). The server computes the value of $Y = \sum_{i\in\mathbb{I}}^{i}(t_i.F_{r_2'}(i))$ and sends the sum of data items (X) to the IoT device along with the hash value of (X, Y) in message M_2. The IoT device needs to construct the key $k_{\mathbb{I}} = \sum_{i\in\mathbb{I}}^{i}(F_{K'}(i).F_{r_2'}(i))$ corresponding to each data item index and compute the value of $Y = X + k_{\mathbb{I}}$ by himself and make sure if the server owns Y by checking the relation $M_2 =?H(..||X||Y||..)$. However, there is no clear logic behind using tag items (t_i) and exploiting this complicated authentication method. There are simpler methods to authenticate the data items $d_i(0 \leq i \leq L)$. For instance, a simple MAC can resolve the issue. In the next section, we propose our enhanced scheme which uses a more efficient method to realize dataset authentication for the IoT device.

Extensive modular exponentiation Modular exponentiation is an expensive discrete-logarithm operation which is so time-consuming for resource-constrained users to perform locally. Some researchers have adopted cloud computing to securely outsource modular exponentiation to cloud servers to reduce computation overhead. Liu's scheme employs 6 modular exponentiation (3 operations by the client and 3 operations by the server) to achieve its desired security goals. Liu et al. have been aware of the large computational overhead of their scheme. They suggested an enhanced version of their basic scheme which employs 4 modular exponentiation (1 operation by the client and 3 operations by the server) and compute the values of u, g^u in advance and store them in the client database. This solution is also unrealistic due to the memory limitation of the client's device. In addition, they truly claim that their enhanced scheme cannot achieve prefect forward secrecy.

Anonymous client authentication and non-revocability In the IoT enviorments, the server is usually in contact with so many different clients. In the authentication phase of Liu et al. 's scheme, it's not clear how the server recognizes which client is his partner. Therefore, in case of the loss or stealing of one device, no revocation mechanism is designed to address this issue.

Privileged inside attack All the clients share the same keys $K^{'}, mk$ with the server. In addition, the server also holds the same data items and tags (d_i, t_i) for the whole clients. Therefore, a hostile client can impersonate the rest of the clients.

4 Our Proposed Scheme

4.1 Initialization phase

When a client c with identity ID_c wants to register to the server **s**, it first generates its signing secret/public key ssk_c/spk_c with a specific K [2] using algorithm 1 $Semecs.Kg$ and submits $ID_c/spk_c, K$ to the server through a secure channel. Here $\lambda = 128$ or $\lambda = 256$. The server chooses the prime value q and initializes the hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ and chooses $mk \in \{0,1\}^\lambda$ as an initial shared key between the IoT device and the server. Then it generates its signing secret/public key ssk_s/spk_s using algorithm 1 $Semecs.Kg$. In addition, the server chooses a random pseudo-identity $TID_c \in \mathbb{Z}_q^*$ for the client and stores the client's real and pseudo-identity (ID_c, TID_c) along with its public key spk_c, K and shared key mk in its own database. In addition, it initializes a session counter $ctn = 0$ for each client. L data items $d_i (0 \leq i \leq L)$ are also stored in the server's database as the second security factor. Finally, it delivers TID_i, mk and $\Omega = \{q, spk_s, L, H, z\}$ to the client in a secure manner. The client also initializes a session counter $ctn = 0$. The long-term secrets of the server and the client are represented as $\mathcal{S}_s = \{(ID_c, TID_c), mk, ssk_s, \mathbb{D} = d_i(0 \leq i \leq L)\}$ and $\mathcal{S}_c = \{mk, ssk_c, TID_c\}$ respectively.

[2] The value K represents how many sessions are required to change the signing keys

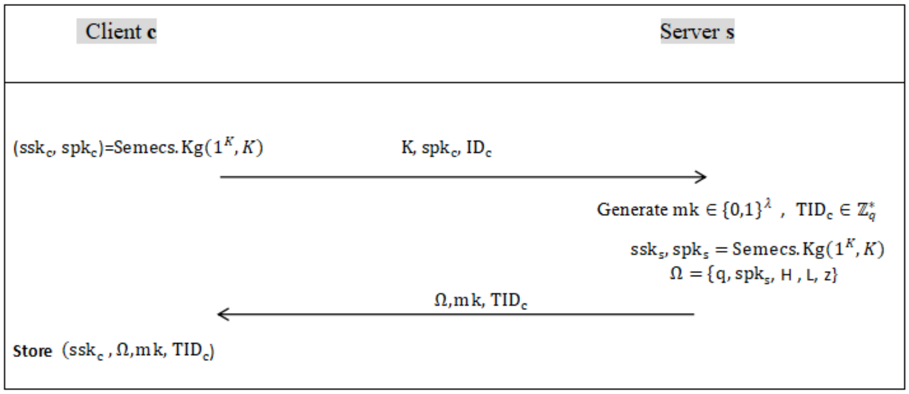

Fig. 3: Liu's registration phase

4.2 Authentication phase

1. At first, the IoT device **c** checks its session counter to see if $ctn \leq K$. Otherwise, it warns the client to register again. Then it sends its pseudo-identity TID_c to the server **s**.

2. After receiving the client's pseudo-identity, the server searches for the TID_c's corresponding record in its database. If the corresponding session counter $ctn \leq K$, it chooses a random subset $\mathbb{I}_s$ of z distinct indices of the records in $\mathbb{D}$ and responds $\mathbb{I}_s$ to the client.

3. The IoT device **c** chooses a random number $r_1 \in \{0,1\}^\lambda$ and computes $R_1 = mk \oplus r_1$. Then it chooses a subset $\mathbb{I}_c$ of z distinct indices of the records in $\mathbb{D}$ disjoint from $\mathbb{I}_s$ ($\mathbb{I}_s \cap \mathbb{I}_c = \emptyset$). Then it sends $R_1, \mathbb{I}_c$ and $M_1 = H(mk||r_1||\mathbb{I}_c||\mathbb{I}_s||TID_c)$ to the server **s**.

4. After receiving the above message, the server **s** calculates $r_1^* = mk \oplus R_1, \mathbb{I} = \mathbb{I}_c \bigcup \mathbb{I}_s$ and checks the equality $M_1 = H(mk||r_1^*||\mathbb{I}_c||\mathbb{I}_s||TID_c)$. If it holds, then the value of $X = \sum_{i\in\mathbb{I}}^{i} H(d_i||ID_c)$ is computed.

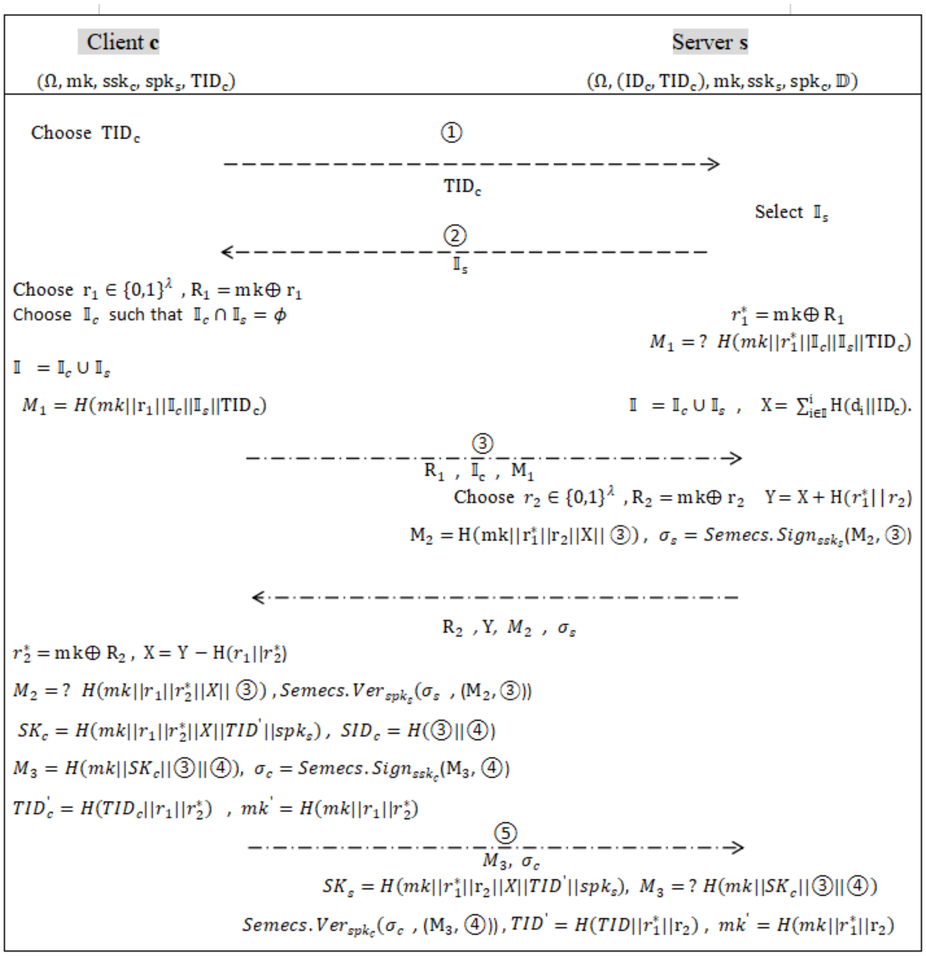

Fig. 4: Our proposed authentication phase

In addition, it chooses the random value $r_2 \in \{0,1\}^\lambda$ and computes $R_2 = mk \oplus r_2, Y = X + H(r_1^*||r_2|) \bmod q, M_2 = H(mk||r_1^*||r_2||X||③)$ and signs the parameters $\sigma_s = Sign(M_2||③)$ using algorithm 1 *Semecs.Sig*. Finally, it sends (Y, σ_s, R_2, M_2) to the IoT device **c**.

5. When the IoT device **c** receives message ④, it computes $r_2^* = mk \oplus R_2, X = Y - H(r_1||r_2^*) \bmod q$ and checks if $M_2 = H(mk||r_1||r_2^*||X||③)$ holds. If this relation holds, it constructs the session key and session identifier as $SK_c = H(mk||r_1||r_2^*||X||TID_c||spk_s)$ and $SID_c = H(③||④)$ respectively. Then, it computes $M_3 = H(mk||SK_c||③||④)$ and signs the parameter M_3 as $\sigma_c = Sign(M_3||④)$ using algorithm 1 *Semecs.Sig* and transmits it to the server **s**. Finally, it updates its pseudo-identity , session counter *ctn* and shared-key as $TID_c^{'} = H(TID_c||r_1||r_2^*), ctn^{'} = ctn{+}1, mk^{'} = H(mk||r_1||r_2^*)$.

6. After receiving the message $\textcircled{5}$, the server **s** computes the session key and session identifier as $SK_s = H(mk||r_1^*||r_2||X||TID_c||spk_s)$ and $SID_s = H(\textcircled{3}||\textcircled{4})$ respectively. Then, it checks the equality $M_3 = H(mk||SK_c||\textcircled{3}||\textcircled{4})$. If this equality holds, it updates client's pseudo-identity , session counter ctn and shared-key in its database as $TID_c^{'} = H(TID_c||||r_1^*||r_2), ctn^{'} = ctn + 1, mk^{'} = H(mk||||r_1^*||r_2)$.

4.3 Revocation phase

In case of the compromise or loss of the IoT device of any client with identity ID_c and pseudo-identity TID_c, the server will remove TID_c, spk_c from the database and disables the attacker to use the network or sign any messages. Then the client ID_c chooses new pseudo-identity TID_c and creates signing secret/public key ssk_s/spk_s using algorithm 1 $Semecs.Kg$ and delivers the updated TID_c, spk_c to the server in a secure manner. Then it stores the updated TID_c, ssk_c in the new IoT device.

5 Security analysis

5.1 Informal analysis

Perfect Forward Secrecy (PFS) In our proposed scheme, even if the public values R_1, R_2, spk_s, TID_c and secret values of both parties $X, ssk_c, ssk_s, mk^{'} = h(mk||||r_1^*||r_2)$ are given to the attacker, based on the one-wayness property of hash function, the attacker has no way to obtain the value mk and compute $r_1 = mk \oplus R_1, r_2 = mk \oplus R_2$ and calculate the previous session keys i.e. $SK = H(mk||r_1^*||r_2||X||TID_c||spk_s)$. Similarly, the nonces r_1, r_2 are protected with the hash function $Y = X + h(r_1, r_2)$. Therefore, compromise of the dataset items (X) doesn't help the attacker to find r_1, r_2.

KCI resilience In the following, we consider two scenarios and show how KCI resilience is achieved in both scenarios.
Scenario I : Server impersonation In the first scenario, $\mathcal{A}$ tries to impersonate the server **s** to the compromised IoT device **c**. Here, $\mathcal{A}$ has access to the secret parameters of the client i.e. $\mathcal{S}_c = \{mk, ssk_c, TID_c\}$. $\mathcal{A}$ needs to construct $M_2 = H(mk||r_1||r_2||X||\textcircled{3})$ and signs the parameters $\sigma = Sign(M_2||\textcircled{3})$ using the secret signing key ssk_s. However, the secret signing key ssk_s of the server isn't accessible to the attacker.

Scenario II : client impersonation

In the second scenario, $\mathcal{A}$ tries to impersonate the client **c** to the compromised server **s**. As a result, it holds long-term credentials of the server i.e. $\mathcal{S}_s = \{(ID_c, TID_c), mk, ssk_s), \mathbb{D}\}$. Here, we give extra capability to the attacker $\mathcal{A}$ and assume that it knows the client's pseudo-identity TID_c and sends it to the server **s** as the first message $\textcircled{1}$. After receiving the server's subset i.e. $\mathbb{I}_s$, it generates the random number $r_1 \in \mathbb{Z}_q^*$ and computes $R_1 = mk \oplus r_1$. Then it chooses a random sub-set of indexes $\mathbb{I}_c$ and constructs $\mathbb{I} = \mathbb{I}_c \bigcup \mathbb{I}_s$. Finally it sends $R_1, \mathbb{I}_c$ and $M_1 = H(mk||r_1||\mathbb{I}_c)$ to the server as the third message ③. The server accepts message ③ and responds with message ④ i.e. $R_2, M_2 = H(mk||r_1||r_2||X||\textcircled{3})$. $\mathcal{A}$ might construct $M_3 = H(mk||SK_c||\mathbb{I}||X||\textcircled{3}||\textcircled{4})$ but fails to sign it and respond $\sigma_c = Sign(M_3||\textcircled{4})$, as the secret signing key of the client isn't available to the attacker $\mathcal{A}$.

Anonymity and unlinkability In our proposed scheme, not only the client's real identity is hidden to the attacker, but also the pseudo-identity and the whole parameters of every session are changed in each session. Therefore, the attacker $\mathcal{A}$ has no way to find any link between the sessions or trace the client.

Revocability In our proposed scheme, each client has a unique identity, pseudo-identity, signing secret/public key. If one of the IoT devices is stolen or lost, the server can recognize the lost device and stop serving it without changing the secrets of the whole network. The revocation mechanism in section 4.3 addresses this issue.

Resistance to privileged inside attack In our proposed scheme, the data-items and client's identities are bound to each other in the $X = \sum_{i \in \mathbb{I}}^{i} H(d_i||ID_c)$ value. In addition, each client holds a separate secret signing key which prevents other hostile clients to impersonate it. Because it can not sign σ_c and respond message ⑤ to the server.

5.2 Formal proof

In this section, we formally prove the session key secrecy and perfect forward secrecy of our proposed scheme using the Real-Or-Random model proposed by Abdalla et al. [20]. Further, we prove the KCI resilience of our proposed scheme using the Bellare and Rogaway model [21].

Participants. Our schemes involve two participants, i.e.: client c and server s. The i-th instance of participant I is denoted by I_i. The i-th instance of c and the j-th instance of s are represented by U_i and S_j, respectively.

Queries. Oracle queries represent the interaction between an adversary $\mathcal{A}$ and the protocol participants. Actually, the adversary capabilities are modelled through queries. The following queries are used by $\mathcal{A}$:

- *Execute(c_i, s_j)* : This query captures the passive eavesdropping of a protocol which outputs all transmitted messages between c_i, s_j.

- *Send(c_i, Start)* The initialization of the protocol is denoted by this query.

- *Send(I_i, m)* : This query simulates an active attacker, $\mathcal{A}$ who can forge message m by manipulation, blocking and intercepting. Then, $\mathcal{A}$ transmits m to instance I_i and receives the response from I_i.

– *Reveal(*(I_i)*)* : This query models the leakage of the I_i's session key SK to $\mathcal{A}$.

– *Corrupt(*I_i*)* : This query shows that $\mathcal{A}$ can compromise either the client if $I_i = c_i$ or the server if $I_i = s_i$. However, it can not compromise both of them simultaneously.

– *Test* (I_i) : This query is used to model the secrecy of the session key generated by I_i. After receiving this query, a binary bit b is chosen such that in case of $b = 0$, a random key with the same size as SK is returned to $\mathcal{A}$. If $b = 1$, SK is given to $\mathcal{A}$. This query can be used any time by the attacker not more than once.

Random oracle: All participants including $\mathcal{A}$ can call the "cryptographic one-way hash function", H(·), which is modeled as a random oracle, say HO.

Partnering: Two instances c_i and s_j are called partners if: (1) c_i and s_j hold the same session identifier (s_{id}), i.e. $S^c_{id} = S^s_{id}$. (2) c_i is the partner identifier(p_{id}) of s_j and vice versa.

Freshness of instance I_i**:** An instance I_i is called fresh, if (1) I_i has completed an accepted session key. (2) $\mathcal{A}$ has used or its partner does not use any *Reveal(*(I_i)*)* query. (3) $\mathcal{A}$ has used *Corrupt(*I_i*)* query no more than once from the beginning of the games.

KCI-Freshness of instance I_i**:** An instance I_i is called KCI-fresh if at the end of the games: (1) I_i has completed an accepted session key. (2) neither *Reveal(*(I_i)*)* nor *Corrupt(*$J \neq I$*)* were performed by the attacker. (3) After issuing *Corrupt(*I*)*, the attacker can no longer issue query *Send(*(J_s, m)*)*, wherein $p_{id}(J_s) = I$.

Definition 2. Semantic security of the session key *As per ROR model,* $\mathcal{A}$ *can break the semantic security of the session key if it can differentiate an actual session key from a random key in a given instance. Let* $Adv^{KS}(\mathcal{A})$ *denote the advantage of the* $\mathcal{A}$ *in breaking session key secrecy of our proposed scheme and* $Succ(A)$ *refers to the event that* $\mathcal{A}$ *uses a* $Test(c)$ *query for some freshly accepted instances, and guesses* b_0 *for the bit* b *that was chosen for the* $Test(c)$*-query. we have* $Adv^{KS}(\mathcal{A}) = |2Pr[Succ(A)] - 1| = |2Pr[b_0 = b] - 1|$*. Our proposed scheme (PS) achieves semantic security of the session key if* $Adv^{KS}(\mathcal{A})$ *is negligible for any PPT attacker.*

Theorem 1. *Assume that a polynomial time adversary* $\mathcal{A}$ *attempts to violate the semantic security of our proposed scheme (PS).* $\mathcal{A}$*'s advantage in breaking the semantic security of our PS is :*

$$Adv^{KS}(\mathcal{A}) \leq \frac{q_h^2}{|H(.)|} \tag{1}$$

Where q_h *and* $|Hash|$ *denote the number of hash queries and range space of hash function* $H(.)$ *, respectively.*

Proof. Our proof is based four sequential games, say $G_i, i \in [0-2]$.

- **Game** G_0: This game simulates a real attacker $\mathcal{A}$ running in random oracle model who has access to all oracles. Thus, we have

$$Adv^{KS}(\mathcal{A}) = |2Pr[E_0] - 1| \tag{2}$$

- **Game** G_1: This game models a passive attack by $\mathcal{A}$ using $Execute(c_i, s_j)$ query. $\mathcal{A}$ can intercept transmitted messages on the public channel. As the session key is constructed as $SK = H(mk||r_1||r_2||X||TID_c||pk_s)$, the attacker needs to compute X, r_1, r_2 to evaluate SK. Eavesdropping messages ① − ⑤ doesn't help attacker $\mathcal{A}$ to this end. Therefore, $\mathcal{A}$ has no additional advantage for wining this game. As a result:

$$Pr[E_1] = Pr[E_0] \tag{3}$$

- **Game** G_2: This game models an active attacker who can use *send* and *hash* queries. The secret parameters of the session key X, r_1, r_2, mk are stored in $M_2 = H(mk||r_1||r_2||X||③||④)$. Using birthday paradox, the maximum probability of finding a collision with q_h queries is $\frac{q'^2_h}{2\times|H(.)|}$. Therefore the advantage of the attacker in this game compared to the game G_1 is:

$$|Pr[E_2] - Pr[E_1]| \leq \frac{q_h^2}{2 \times |H(.)|} \tag{4}$$

  In order to win the game G_2 after execution of the *Test* query, $\mathcal{A}$ needs to guess the bit c' with maximum probability of $\frac{1}{2}$:

$$|Pr[E_2]| = \frac{1}{2} \tag{5}$$

  Using equations (3) , (4) and (5) we have :

$$\frac{1}{2}.Adv^{KS}(\mathcal{A}) = |Pr[E_0] - \frac{1}{2}| = |Pr[E_1] - Pr[E_2]| \tag{6}$$

  Therefore:

$$Adv^{KS}(\mathcal{A}) \leq \frac{q_h^2}{|H(.)|} \tag{7}$$

Definition 3. Perfect Forward Secrecy (PFS) in ROR model *Let $Adv^{PFS}(\mathcal{A})$ denote the advantage of the $\mathcal{A}$ in breaking perfect forward key secrecy of our proposed scheme and $Succ(A)$ refers to the event that $\mathcal{A}$ uses a $Test(c)$ query for some freshly accepted instances. Then it issues $Corrupt(c), Corrupt(s)$ queries and guesses b_0 for the bit b that was chosen for the $Test(c)$ query. We have $Adv^{PFS}(\mathcal{A}) = |2Pr[Succ(A)] - 1| = |2Pr[b_0 = b] - 1|$. Our proposed scheme (PS) achieves PFS if $Adv^{PFS}_{\mathcal{A}}(t)$ is negligible for any PPT attacker.*

Theorem 2. *Assume that a polynomial time adversary $\mathcal{A}$ attempts to violate perfect forward security of our proposed scheme (PS). $\mathcal{A}$'s advantage in breaking the PFS of our PS is:*

$$Adv^{PFS}(\mathcal{A}) \leq \frac{q_h^2}{|H(.)|} \tag{8}$$

Proof. . Our proof is based on previous games, i.e. $G_i, i \in [0-2]$. In order to model the corruption of the client and server, we use the following game G_3, G_4:

Game G_3: As an extension to G_2, this game uses $Corrupt(c)$ query to simulate the stolen IoT device attack with side-channel techniques. Here, the attacker can obtain $mk^{'} = h(mk||||r_1||r_2)$, however the one-wayness property of hash function prevents him to compute mk which is essential to compute r_1, r_2. Therefore, games G_3, G_2 are identical except for finding collision on $mk^{'}$ using $q_h^{'}$ queries. Therefore, using birthday paradox we have:

$$|Pr[E_3] - Pr[E_2]| \leq \frac{{q_h^{'}}^2}{2 \times |H(.)|} \tag{9}$$

Game G_4: This game simulates the corruption of the server using $Corrupt(s)$ query. Compared to the game G_3, the advantage of the attacker is the secret data-items of the server. Therefore, the secret value of X can be computed by the attacker. However, he can only obtain the hash value of the nonces r_1, r_2 with this secret i.e. $h(r_1, r_2) = Y - X$, which is not useful to obtain the secret keys. Using birthday paradox, the advantage of the attacker is:

$$|Pr[E_4] - Pr[E_3]| \leq \frac{{q_h^{''}}^2}{2 \times |H(.)|} \tag{10}$$

Winning the game G_4, requires $\mathcal{A}$ to guess the bit $c^{'}$ with maximum probability of $\frac{1}{2}$:

$$|Pr[E_4]| = \frac{1}{2} \tag{11}$$

The advantage of the attacker to violate PFS of our proposed scheme is

$$Adv^{PFS}(\mathcal{A}) = |2Pr[E_0] - 1| \tag{12}$$

Using triangular inequality on equations (3) , (9-11), we have:

$$\frac{1}{2}.Adv^{PFS}(\mathcal{A}) = |Pr[E_0] - \frac{1}{2}| = |Pr[E_1] - Pr[E_4]| \leq \frac{q_h^2}{2 \times |H(.)|} \Longrightarrow Adv^{PFS}(\mathcal{A}) \leq \frac{q_h^2}{|H(.)|} \tag{13}$$

Where $q_h^2 = {q_h^{'}}^2 + {q_h^{''}}^2$.

Definition 4. Proveable KCI-security *Let $Adv_I^{KCI}(\mathcal{A})$ denote the advantage of the $\mathcal{A}$ in impersonating the party $J \neq I$ to the corrupted I participant. Also $Succ(A)$ refers to the event that $\mathcal{A}$ uses a $Test(c)$ query for some KCI-freshly accepted instances I_i, and guesses b_0 for the bit b that was used for the $Test(c)$-query. We have $Adv_I^{KCI}(\mathcal{A}) = |2Pr[Succ(A)] - 1| = |2Pr[b_0 = b] - 1|$. Our PS achieves KCI secrecy against corruption of I if $Adv_I^{KCI}(\mathcal{A})$ is negligible for any PPT attacker.*

Theorem 3. *If $Adv_c^{KCI}(\mathcal{A})$ denotes the advantage of the attacker $\mathcal{A}$ in breaking KCI secrecy against corruption of $I = c$ and $Adv_s^{KCI}(\mathcal{A})$ represents the advantage of the attacker $\mathcal{A}$ in breaking KCI secrecy against corruption of $I = s$, we have:*

$$Adv_c^{KCI}(\mathcal{A}) = Adv_s^{KCI}(\mathcal{A}) = |2Pr[E_0] - 1| \leq \frac{q_h^2}{|H(.)|} + \epsilon_{DLP} \tag{14}$$

Proof. . In order to prove theorem 3, we consider two scenarios: corruption of the client c and corruption of the server s. Our proof is based on previous games G_0, G_1, G_2. In order to model the corruption of the client c in the first scenario or corruption of the server s in the second scenario, we use the following game:

Game G_3: This game corresponds to the corruption of the client c through $Corrupt(c)$ query in the first scenario or corruption of the server s using $Corrupt(s)$ query in the second scenario. In the first scenario, the attacker obtains the secrets of the client c i.e. mk, ssk_c. Although, it can construct message M_2 using fake data-items X, it fails to sign message M_2, as the secret signing key of the server ssk_s is not leaked to the attacker $\mathcal{A}$. The only way to forge the signature of the server is to solve DLP problem with probability ϵ_{DLP} to obtain the secret signing key from the signing public key. In the second scenario, the attacker obtains the secrets of the server s i.e. mk, X, ssk_s. Although, it can compute message $M_3 = H(mk||SK_c||\mathbb{I}||X||\textcircled{3}||\textcircled{4})$, but signing this message requires the secret signing key of the client which is hidden to the attacker $\mathcal{A}$, unless it solves DLP problem with probability ϵ_{DLP}.

Therefore, we have:

$$|Pr[E_3] - Pr[E_2]| \leq \epsilon_{DLP} \tag{15}$$

Based on the games $G_i, i \in [0-3]$, the advantage of the attacker to violate KCI of our proposed scheme in both scenarios $Adv_c^{KS}(\mathcal{A}), Adv_s^{KS}(\mathcal{A})$ is:

$$Adv_c^{KS}(\mathcal{A}) = Adv_s^{KS}(\mathcal{A}) = |2Pr[E_0] - 1| \leq \frac{q_h^2}{|H(.)|} + \epsilon_{DLP} \tag{16}$$

6 Comparative Summary: Security and performance

To demonstrate the security and usability of our proposed scheme, we provide a comparative measurement on the security, computation and communication cost of recent big-data based schemes (Chan et al. and Liu et al.) and our proposed scheme.

6.1 Security comparison

In order to perform an objective security comparison, we use the evaluation metrics introduced in section 2.3. The comparison results are depicted in Table 3.

Chan et al.'s scheme as the first big-data based scheme, only achieves authentication of the server to the client, but not vice versa. In addition, it's not truly two-factor secure assuming a strong adversary. Other security requirements are also not satisfied. On the other hand, Liu et al.'s scheme achieves mutual authentication between the server and the IoT device, but their scheme isn't truly two-factor secure and suffers serious vulnerabilities such as KCI attack, inside attack and non-revocability assuming a strong attacker. Our proposed scheme can achieves all the desired properties assuming a strong attacker who can compromise the whole secrets of the server.

Table 2: Comparison on security requirements in recent big data-based schemes

Big data based Schemes	**Security Requirement**									**Threat Model**
	SR_1	SR_2	SR_3	SR_4	SR_5	SR_6	SR_7	SR_8	SR_9	
Chan et al.	×	×	×	×	×	×	×	×	✓	Weak
Liu et al.	✓	✓	×	×	×	✓	×	✓	✓	Weak
Our proposed scheme	✓	✓	✓	✓	✓	✓	✓	✓	✓	Strong

Note: SR_1 : Mutual authentication , SR_2 : Key Agreement with Secrecy , SR_3 : Two-factor security , SR_4 : Revocability , SR_5 : Anonymity and unlinkability , SR_6 : (Perfect) forward secrecy , SR_7 : Key Compromise Impersonation (KCI) resilience , SR_8 : Resistance to privileged inside attack, SR_9 : Resistance to well-known attacks:

6.2 Computational comparison

In order to compare the computational costs of our proposed scheme with previous relevant schemes, we implement our scheme and report the running time of each operation. To make our comparison more accurate, we used Liu et al.'s framework which consists of a PC with Intel® Core™ i7-4770 CPU@ 3.4 GHz processor with 16 GB RAM as the server and a Raspberry Pi 3 Model B+ with ARM Cortex-A53 @ 1.4 GHz processor and 1 GB RAM as the IoT device. Similarly, for security levels $\lambda = 128$ and $\lambda = 256$, we used the Koblitz curve secp256k1 and secp521r1, respectively. For the hash function H, we use SHA-256.

In Tables 4 and 5, we compare the number of different types of computations in our scheme compared to previous schemes for the IoT device and server respectively.

Table 3: Execution time of cryptographic primitives performed by IoT device

IoT device

	Modular Exp	**Mult**	**Add**	**PRF E**	**PRF F**	**Hash H**	**Total (ms)**
$\lambda = 128$, Elliptic Curve: secp256k1, Data item size = 0.00005 MB , z=50							
Liu et al.'s scheme	3 (13.6 ms)	2z (0.16 ms)	2z (0.03 ms)	1 (0.04 ms)	4z (0.66 ms)	5 (0.06 ms)	15.09
Our scheme	2 (9.03 ms)	2 (0.003 ms)	2 (0.0006 ms)	–	–	13 (0.15 ms)	9.13
$\lambda = 256$, Elliptic Curve: secp521r1, Data item size = 0.00005 MB , z=50							
Liu et al.'s scheme	3 (81.96 ms)	2z (0.34)	2z (0.05 ms)	1 (0.04 ms)	4z (0.96)	5 (0.12 ms)	84.73
Our scheme	2 (54.6 ms)	2 (0.006 ms)	2 (0.001 ms)	–	–	13 (0.31 ms)	54.91

Our scheme has reduced modular exponentiation operation and multiplication for the both the IoT and the server compared to Liu et al.'s scheme which causes significant reduction of the execution time. PRF E and PRF F are also no longer required. For the IoT device, one third of the execution time is reduced i.e 6 ms and 30 ms for security levels of $\lambda = 128, \lambda = 256$ respectively. The execution time of the server is also much reduced i.e 0.3 ms and 4 ms for security levels of $\lambda = 128, \lambda = 256$ respectively.

Table 4: Execution time of cryptographic primitives performed by server

Server

	Modular Exp	**Mult**	**Add**	**PRF E**	**PRF F**	**Hash H**	**Total (ms)**
$\lambda = 128$, Elliptic Curve: secp256k1, Data item size = 0.00005 MB , z=50							
Liu et al.'s scheme	3 (1.69 ms)	4z+1 (0.04 ms)	4z-2 (0.01 ms)	1 (0.04 ms)	2z (0.6 ms)	2z+5 (0.37 ms)	2.44
Our scheme	2 (1.12 ms)	2 (0.02 ms)	z+2 (0.005 ms)	–	–	z+7 (0.37 ms)	2.15
$\lambda = 256$, Elliptic Curve: secp521r1, Data item size = 0.00005 MB , z=50							
Liu et al.'s scheme	3 (11.08 ms)	4z+1 (0.09 ms)	4z-2 (0.02 ms)	1 (0.02 ms)	2z (0.7 ms)	2z+5 (0.73 ms)	12.51
Our scheme	2 (7.38 ms)	2z+1 (0.04 ms)	2z-2 (0.01 ms)	1 (0.02 ms)	2z (0.07 ms)	2z+6 (0.73 ms)	8.25

7 Conclusion

Using Big-data as a novel authentication factor for IoT was first initiated by Chan et al.'s and later improved by Liu et al., who added novel security properties such as PFS, KCI and SCI resilience. In this paper, we showed that by assuming a real strong attacker, KCI and SCI resilience don't hold anymore. In addition, their scheme suffers inside attack and doesn't provide revocability. Then we proposed our novel authentication scheme which provides PFS, KCI resilience, revocability, inside attack resilience in a real attacker setting. Further, our performance analysis shows that our scheme has reduced modular exponentiation operation and multiplication for both the IoT device server compared to Liu et al.'s scheme. Therefore, the running time of both IoT device and the server is reduced by one third.

References

1. A. C.-F. Chan, J. W. Wong, J. Zhou, and J. Teo, "Scalable two-factor authentication using historical data," in *European Symposium on Research in Computer Security*, pp. 91–110, Springer, 2016.
2. B. Liu, Q. Tang, and J. Zhou, "Bigdata-facilitated two-party authenticated key exchange for iot," in *International Conference on Information Security*, pp. 95–116, Springer, 2021.
3. Y. Zhang, H. Xian, and A. Yu, "Csnn: Password guessing method based on chinese syllables and neural network," *Peer-to-Peer Networking and Applications*, vol. 13, no. 6, pp. 2237–2250, 2020.
4. M. K. Sharma and M. J. Nene, "Dual factor third-party biometric-based authentication scheme using quantum one time passwords," *Security and Privacy*, vol. 3, no. 6, p. e129, 2020.
5. T. Sabhanayagam, V. P. Venkatesan, and K. Senthamaraikannan, "A comprehensive survey on various biometric systems," *International Journal of Applied Engineering Research*, vol. 13, no. 5, pp. 2276–2297, 2018.
6. "Hackers Unlock Any Phone Using Photographed Fingerprints In Just 20 Minutes, howpublished = `https://www.tomsguide.com/news/hackers-unlock-any-phone-using-photographed-fingerprints-in-just-20-minutes`, note = Accessed: 2019-10-10."
7. Q. Jiang, J. Ma, G. Li, and L. Yang, "An enhanced authentication scheme with privacy preservation for roaming service in global mobility networks," *Wireless personal communications*, vol. 68, no. 4, pp. 1477–1491, 2013.
8. F. Wen, W. Susilo, and G. Yang, "A secure and effective anonymous user authentication scheme for roaming service in global mobility networks," *Wireless personal communications*, vol. 73, no. 3, pp. 993–1004, 2013.
9. P. Gope and T. Hwang, "Enhanced secure mutual authentication and key agreement scheme preserving user anonymity in global mobile networks," *Wireless Personal Communications*, vol. 82, no. 4, pp. 2231–2245, 2015.
10. M. N. Aman, K. C. Chua, and B. Sikdar, "Mutual authentication in iot systems using physical unclonable functions," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1327–1340, 2017.

11. U. Chatterjee, V. Govindan, R. Sadhukhan, D. Mukhopadhyay, R. S. Chakraborty, D. Mahata, and M. M. Prabhu, "Building puf based authentication and key exchange protocol for iot without explicit crps in verifier database," *IEEE transactions on dependable and secure computing*, vol. 16, no. 3, pp. 424–437, 2018.
12. K. B. Frikken, M. Blanton, and M. J. Atallah, "Robust authentication using physically unclonable functions," in *International Conference on Information Security*, pp. 262–277, Springer, 2009.
13. P. Gope and B. Sikdar, "Lightweight and privacy-preserving two-factor authentication scheme for iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580–589, 2018.
14. Q. Jiang, X. Zhang, N. Zhang, Y. Tian, X. Ma, and J. Ma, "Two-factor authentication protocol using physical unclonable function for iov," in *2019 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 195–200, IEEE, 2019.
15. P. Gope, J. Lee, and T. Q. Quek, "Lightweight and practical anonymous authentication protocol for rfid systems using physically unclonable functions," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2831–2843, 2018.
16. J. Brainard, A. Juels, R. L. Rivest, M. Szydlo, and M. Yung, "Fourth-factor authentication: somebody you know," in *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 168–178, 2006.
17. S. Choi and D. Zage, "Addressing insider threat using "where you are" as fourth factor authentication," in *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*, pp. 147–153, IEEE, 2012.
18. A. A. Yavuz and M. O. Ozmen, "Ultra lightweight multiple-time digital signature for the internet of things devices," *IEEE Transactions on Services Computing*, 2019.
19. M. A. Strangio, "On the resilience of key agreement protocols to key compromise impersonation," in *European Public Key Infrastructure Workshop*, pp. 233–247, Springer, 2006.
20. M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," *IEE Proceedings-Information Security*, vol. 153, no. 1, pp. 27–39, 2006.
21. M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Annual international cryptology conference*, pp. 232–249, Springer, 1993.