symmetry

MDPI

Article

A Strong Designated Verifier Proxy Re-Signature Scheme for IoT Environments

Xiao-Dong Yang *, Li-Kun Xiao, Chun-Lin Chen and Cai-Fen Wang

College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China; xiaolkun@163.com (L.-K.X.); chenchlgs@163.com (C.-L.C.); wangcfen@126.com (C.-F.W.)

* Correspondence: y200888@163.com

Received: 7 September 2018; Accepted: 13 October 2018; Published: 2 November 2018

Abstract: With the rapid popularization of the Internet of Things (IoT) in our daily lives, the communication security and identity privacy of IoT devices must be ensured. However, traditional authentication mechanisms utilized in IoT cannot completely ensure a user's privacy when his/her messages are routed via an untrusted intermediate device. Strong designated-verifier proxy re-signature (SDVPRS) is a new cryptographic technology that combines the advantages of strong designated verifier signature and proxy re-signature. Therefore, SDVPRS is considered to be a better approach to maintain data integrity and protect the identity privacy of the signer in a resource-limited IoT device. Nevertheless, designing a secure SDVPRS scheme without random oracles is still a challenging task. In this paper, we mainly focus on such a construction by providing a new method. We first provide the formal definition of SDVPRS and its security model. Then, we present the first SDVPRS scheme, which is bidirectional, multi-use and non-transferable, and we prove its security under the standard complexity assumptions in the standard model. The analysis results show that our SDVPRS scheme can not only protect the privacy of the signer's identity, but also provide non-delegatability for signature verification. We present an example of potential application to environmental monitoring systems using our SDVPRS scheme.

Keywords: proxy re-signature; strong designated verifier signature; standard model; identity privacy; data integrity

1. Introduction

The Internet of Things (IoT) is rapidly entering all aspects of our daily lives. IoT uses sensors, radio frequency identification (RFID), wireless data communications and other technologies to construct a network that covers all things in the world to make the interactions between people and things and between things and things more intelligent and convenient [1]. IoT devices equipped with sensors have the ability to sense and process information, and they are used to collect, transmit and disseminate data from the field to a server or other IoT devices. IoT has been deployed in many environments, such as smart transportation, smart cities, environmental surveillance, smart homes, military target tracking, biomedical health monitoring and industrial automation [2].

IoT is everywhere in our daily lives and offers great benefits for human life. However, IoT data are transmitted over public networks, and ensuring a user's privacy and data security is of particular importance [3]. In an IoT environment, most IoT devices have limitations in terms of energy capacity, storage capacity and computing power. Therefore, conventional cryptosystems cannot be implemented in resource-constrained IoT devices. Digital signature technology guarantees the integrity of the data during transmission and also authenticates the identity of the sender. Based on various digital signature techniques, such as identity-based signature and certificateless signature, researchers have proposed several schemes [2–6] to ensure the integrity and authenticity of IoT data transmitted over public channels. However, anyone can use the signer's public key to verify the validity of the signature, so

these schemes [2–6] reveal some private information of the signer (such as the identity of the signer). Strong designated verifier proxy re-signature (SDVPRS) can provide a better solution to solve these problems. Due to the combination of the features of strong designated verifier signature (SDVS) and proxy re-signature (PRS), SDVPRS not only maintains the integrity of IoT data during transmission, but also protects the identity privacy of the IoT device that signs IoT data. In an SDVPRS scheme, only the designated verifier can verify the validity of a signature. Specifically, SDVPRS allows a signer to designate a verifier, and a semi-trusted proxy is allowed to convert the signer or the designated verifier into a signature. However, how to design an SDVPRS scheme without random oracles is still challenging. Hence, we focus on constructing an SDVPRS scheme in the standard model that can be applied to IoT devices.

PRS, which allows a proxy to transform the signature of a message generated by Alice into Bob's signature of the same message, is an important cryptosystem in cryptography [7]. However, the proxy by itself is unable to create arbitrary signatures on behalf of Alice or Bob. If the proxy can not only convert Alice's signature into Bob's signature, but also convert Bob's signature into Alice's signature, then we say that a PRS scheme is bidirectional. Additionally, if the transformed signatures can be further transformed by the proxy, then we say that a PRS scheme is multi-use. Since PRS can convert signatures, it has been applied to key management, cross-domain identity authentication and other fields [7–9].

To designate a verifier to verify the validity of a signature, the designated verifier signature (DVS) was presented by Jakobsson et al. [10]. A DVS scheme ensures that only the designated verifier can verify signatures generated by the signer. However, the designated verifier is able to produce simulated signatures that are computationally indistinguishable from the real signatures on the same messages created by the signer. Consequently, a DVS scheme provides the authentication of signatures, but it does not satisfy the non-repudiation of ordinary signatures since only the designated verifier is able to ensure that signatures are generated by a real signer. Specifically, in a DVS scheme, both the signer and the designated verifier can generate a valid signature of the message. To avoid man-in-the-middle attacks, Jakobsson et al. [10] further proposed the concept of the strong designated verifier signature (SDVS), which requires the secret key of the designated verifier during signature verification. Since the attacker does not know the verifier's secrete key, the validity of the intercepted signature cannot be verified. SDVS has stronger security and many special applications such as voting and deniable authentication [11–13].

Based on the concepts of PRS and DVS, designated verifier proxy re-signature (DVPRS) was introduced by Wei et al. [14]. In a DVPRS scheme, a semi-trusted proxy can change the signer or the verifier in a DVS. Consequently, DVPRS has the properties of both PRS and DVS. DVPRS is a useful technique in deniable or anonymous authentication, and it can be applied to fields such as wireless communication networks [15–17].

To enhance the privacy of the signer's identity, we introduce the concept of SDVPRS in this paper, which is a variant of DVPRS. In SDVPRS, the designated verifier's secret key is required in the signature verification; thus, the validity of the signature can only be checked by the signer or the designated verifier. This approach ensures that an adversary who captures a signature only knows that either the signer or the designated verifier created the signature, but the adversary cannot infer which one of them is the real signature generator. No third party other than the designated verifier knows the true identity of the signer; thus, SDVPRS can protect the privacy of the signer's identity. Because SDVPRS combines the advantages of SDVS and PRS, most of the significant security requirements, such as integrity, unforgeability and non-transferability, and the signer's identity privacy protection can be guaranteed in a single logic step.

The security concepts of DVPRS were presented by Wei et al. [14], but the formal definition of SDVPRS was not taken into account in [14]. In addition, Wei et al. [14] designed a DVPRS scheme in the random oracle model, which is also the only publicly available DVPRS scheme. Unfortunately, the random oracle model might not ensure the security of the scheme if the random oracles are instantiated

with concrete hash functions [18]. In fact, Wei et al.'s proposal [14] is an SDVPRS scheme since the verification of the signature requires the designated verifier's secret key. Therefore, constructing a secure (S)DVPRS scheme without random oracles in the standard model is an unsolved problem.

1.1. Our Contributions

In this paper, we first present the security concepts of SDVPRS. In contrast to PRS, our formal definition for SDVPRS relies on the security concepts of SDVS. Based on Waters' technique [19], we then present a construction of an SDVPRS scheme without random oracles, which is proven to be existentially unforgeable in the standard model. Our SDVPRS scheme is bidirectional, multi-use, transparent and non-transferable. Furthermore, the security proof shows that our SDVPRS scheme can ensure the integrity and authenticity of IoT data, as well as protect the identity privacy of the IoT device. To the best of our knowledge, our proposal is the first (strong) designated verifier PRS scheme without random oracles. Our SDVPRS scheme is very useful for protecting the security of IoT data and the identity privacy of the sender.

1.2. Related Work

With the substantial development of cloud computing and IoT techniques, data privacy [20–23], access control [24,25] and message authentication [26–28] have become important issues and the focus of many studies. According to the diverse requirements of the authentication, various signature schemes, such as homomorphic signature schemes [29,30] and proxy signature schemes [31–33], have been proposed. However, most of the existing message authentication schemes do not consider the privacy of IoT devices. SDVPRS is a new cryptographic technology that has the advantages of PRS and SDVS, so we introduce SDVPRS to solve the sender's identity privacy problem in IoT environments.

The concept of PRS was presented by Blaze et al. [7] in 1998, and the security definition of PRS was formalized by Ateniese and Hohenberge [34] in 2005. Since then, researchers have designed a large number of PRS schemes with special properties. Hu et al. [35] proposed a secure identity-based proxy re-signature scheme under the standard model, but its security relies on strong difficult problem assumptions. Tian [36] designed an identity-based proxy re-signature scheme over lattices, but the size of the signature and secret key was relatively large. Wang and Xia [37] presented an identity-based proxy re-signature scheme with the aggregate property. However, their scheme required numerous system parameters. To reduce the security risks of an individual proxy, Yang et al. [38] introduced the concept of threshold proxy re-signature, which can distribute the re-signature key to multiple proxies for management. Yang et al. [39] introduced the concept of flexible threshold proxy re-signature, which can flexibly select different thresholds according to the importance of the message to be re-signed. To improve the response time of re-signing, Yang et al. [40] proposed an on-line/off-line threshold proxy re-signature scheme, which completes most of the computational tasks of re-signing in the off-line phase. To solve the key escrow problem in identity-based proxy re-signature, some certificateless proxy re-signature schemes [41,42] have been proposed. Unfortunately, these schemes have some security flaws [43].

The first DVS scheme was presented by Jakobsson et al. [10] in 1996. Saeednia et al. [44] gave the formal definition of SDVS. Later, some SDVS schemes were presented in [45–47]. Hung et al. [48] designed a secure SDVS scheme in the standard model, but its security depended on the security of pseudo-random functions. Hence, their scheme has potential security risks. Based on the standard complexity assumptions, Tian et al. [49] designed two SDVS schemes without random oracles. The researchers also proposed some variants of DVS such as universal DVS [50,51] and multi-verifier DVS [52,53]. Until now, the only DVPRS scheme was that proposed by Wei et al. [14], but its security was dependent on ideal random oracles.

Data security has become an important issue in IoT. When IoT data are transmitted through open and insecure channels, they are vulnerable to various attacks, such as forgery attack, tamper attack, and so on. To ensure the communication security of IoT devices, Jia et al. [3] proposed a data authentication

scheme based on a certificateless signature. Combining aggregate signature and identity-based signature, Shen et al. [2] designed a data integrity protection scheme for wireless sensor networks. Kumar et al. [4] proposed a secure data transmission scheme for a healthcare wireless sensor network using certificateless aggregation signature technology. Yeh et al. [6] proposed an efficient certificateless signature scheme to ensure the security of IoT devices. However, these schemes [2–6] were proven to be secure in the random oracle model, which means that these schemes might be insecure in reality. In particular, these existing schemes protect the integrity of IoT data, but at the same time, they disclose the identity privacy of IoT devices. Motivated by this scenario, we construct an SDVPRS scheme in the standard model to protect the integrity of IoT data and the privacy of the sender's identity. The proxy converts the IoT device's signature to a group's signature on the same data, thereby reducing the risk of identifying the identity of the IoT device according to the signature and realizing the anonymity of data transmission. In addition to the designated verifier, no one can verify the legality of the final signature. That is, our scheme enables the integrity and authenticity of IoT data to be verified without revealing the user's identity privacy.

2. Preliminaries

2.1. Bilinear Pairing

Assume that p is a large prime, G_1 and G_2 are two multiplicative cyclic groups of order p and g is an arbitrary generator of G_1. A map $e : G_1 \times G_1 \rightarrow G_2$ is called a bilinear pairing if it satisfies the following conditions.

- Bilinearity: $e(g^x, g^y)$=$e(g, g)^{xy}$, where $x, y \in Z_p$.
- Non-degeneracy: $e(g, g) \neq 1_{G_2}$, where 1_{G_2} is the identity element of G_2.
- Computability: $e(g^x, g^y)$ is efficiently computable, where $x, y \in Z_p$.

2.2. Complexity Assumptions

Polynomial-time algorithms are unable to solve the following hard problems [34,54], which are considered to be intractable in complexity theory.

Definition 1. *Given four elements $g, g^a, g^b, g^c \in G_1$, where unknown values a, b and c are randomly selected from Z_p, the bilinear Diffie–Hellman (BDH) problem in (G_1, G_2) is to calculate $e(g, g)^{abc} \in G_2$.*

Definition 2. *Given $g, g^a, g^b, g^c \in G_1$ and $Z \in G_2$ where unknown values a, b and c are randomly selected from Z_p, the decisional bilinear Diffie–Hellman (DBDH) problem in (G_1, G_2) is to determine whether $Z = e(g, g)^{abc}$ holds.*

Taking as input $(g, g^a, g^b, g^c) \in G_1^4$ and $Z \in G_2$, the DBDH oracle $\mathcal{O}_{DBDH}$ outputs one if $Z = e(g, g)^{abc}$; else, it outputs zero.

Definition 3. *Given $g, g^a, g^b, g^c \in G_1$ where unknown values a, b and c are randomly selected from Z_p, the gap bilinear Diffie–Hellman (GBDH) problem in (G_1, G_2) is to use the oracle $\mathcal{O}_{DBDH}$ to calculate $e(g, g)^{abc} \in G_2$.*

The main difference between the BDH problem and the GBDH problem is whether the DBDH oracle $\mathcal{O}_{DBDH}$ is required to solve the corresponding problem.

3. Security Model and System Framework

3.1. The Syntax of SDVPRS

An SDVPRS scheme includes the following nine algorithms:

- **Setup**: This algorithm takes a security parameter $\lambda \in Z$ as input and produces system parameters sp.
- **KeyGen**: Upon input of sp, this algorithm outputs a secret key sk and a corresponding public key pk.

- **ReSKey**: Upon input of sp, a signer's key pair (pk_A, sk_A) and another signer's key pair (pk_B, sk_B), this algorithm generates a re-signing key $rsk_{A\to B}$ for the proxy.
- **ReVKey**: Upon input of sp and two verifiers' key pairs (pk_C, sk_C) and (pk_D, sk_D), this algorithm generates a re-designate-verifier key $rvk_{C\to D}$.
- **Sign**: This algorithm takes sp, a signer's secret key sk_S, a message m and a designated verifier's public key pk_V as input. It outputs a signature σ on m.
- **ReSign**: This algorithm takes sp, a re-signing key $rsk_{A\to B}$ between a signer S_A and another signer S_B and a signature σ_{AC} on a message m under the signer S_A and a verifier V_C as input. It generates a re-signature σ_{BC} on m under S_B and V_C.
- **ReVer**: This algorithm takes sp, a re-designate-verifier key $rvk_{C\to D}$ between a verifier V_C and another verifier V_D and a signature σ_{AC} on a message m under a signer S_A and the verifier V_C as input. It generates a re-signature σ_{AD} on m under S_A and V_D.
- **Verify**: This algorithm takes sp, a signer's public key pk_S, a designated verifier's secret key sk_V and a signature σ on a message m as input. It outputs one if σ is a valid signature; otherwise, it outputs zero.
- **Sim**: This algorithm takes sp, a signer's public key pk_S, a designated verifier's secret key sk_V and a message m as input. It generates a simulated signature σ' that is indistinguishable from the one created by the signer.

3.2. Security Model of SDVPRS

An SDVPRS scheme consists of three entities: the signer, the proxy and the designated verifier. The security model of SDVPRS mainly considers the following four security concepts. Among them, two properties, unforgeability and non-delegatability for signature verification, ensure the integrity and authenticity of the IoT data in the communication process, and the other two properties, non-transferability and privacy of the signer's identity (PSI), prevent any third party from obtaining the identity information of the IoT device from a signature. Similar to the security model of a bidirectional PRS scheme [34], the security model of a bidirectional SDVPRS scheme also requires that the proxy is semi-trusted and is not allowed to collude with the signer or the designated verifier.

Unforgeability means that a legal signature can only be generated by the signer or the designated verifier. We define a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ to describe the unforgeability of an SDVPRS scheme.

- **Setup**: $\mathcal{C}$ executes the algorithms **Setup** and **KeyGen** to generate system parameters sp, the key pair (pk_S, sk_S) of the target signer and the key pair (pk_V, sk_V) of the target verifier. Then, $\mathcal{C}$ sends (sp, pk_S, pk_V) and the public keys of other users to $\mathcal{A}$.
- **Queries**: $\mathcal{A}$ may adaptively request the following oracles built by $\mathcal{C}$.
  - $\mathcal{O}_{Sign}$: Upon input of message m_i, this oracle outputs a signature $\sigma_i = \textbf{Sign}(sk_S, pk_V, m_i)$ on m_i.
  - $\mathcal{O}_{ReSKey}$: Upon input of two signers' public keys pk_i and pk_j, this oracle outputs a re-signing key $rsk_{i\to j} = \textbf{ReSKey}(pk_i, sk_i, pk_j, sk_j)$, where sk_i and sk_j are the secret keys corresponding to pk_i and pk_j, respectively.
  - $\mathcal{O}_{ReVKey}$: Upon input of two verifiers' public keys pk_i and pk_j, this oracle outputs a re-designate-verifier key $rvk_{i\to j} = \textbf{ReVKey}(pk_i, sk_i, pk_j, sk_j)$.
  - $\mathcal{O}_{Sim}$: Upon input of a message m_i, this oracle returns a simulated signature $\sigma_i' = \textbf{Sim}(sk_V, pk_S, m_i)$ on m_i.
  - $\mathcal{O}_{Verify}$: This oracle takes a message m_i and a signature σ_i as input, and it outputs a decision $dec = \textbf{Verify}(sk_V, pk_S, m_i, \sigma_i)$, where $dec \in \{0, 1\}$.
- **Forgery**: Eventually, $\mathcal{A}$ produces a forgery (m^*, σ^*). The adversary $\mathcal{A}$ wins if the following conditions are satisfied.
  1. $\textbf{Verify}(sk_V, pk_S, m^*, \sigma^*)$=1, which means that σ^* is a valid signature on m^*.
  2. m^* has never been submitted to $\mathcal{O}_{Sign}$ and $\mathcal{O}_{Sim}$.

Definition 4. *If there is no polynomial-time attacker $\mathcal{A}$ who can win in the above game with a non-negligible probability, then we say that a bidirectional SDVPRS scheme is existentially unforgeable against adaptive chosen-message attacks.*

The property of non-transferability means that any third party cannot distinguish whether the real generator of a signature is the signer or the designated verifier.

Definition 5. *If the signature created by the signer and the signature simulated by the designated verifier are computationally indistinguishable, then we say that an SDVPRS scheme is non-transferable [14].*

PSI ensures that no one other than the designated verifier can infer the signer's true identity from a signature. Specifically, two signers S_0 and S_1 produce signatures for a designated verifier V. Given a signature σ on a message m, anyone without V's secret key is unable to determine whether σ is created by S_0 or S_1. We provide a game between a distinguisher $\mathcal{D}$ and a challenger $\mathcal{B}$ to describe the formal definition of PSI.

- **Setup**: $\mathcal{B}$ performs the algorithms **Setup** and **KeyGen** to generate system parameters sp, S_0's key pair (pk_{S_0}, sk_{S_0}), S_1's key pair (pk_{S_1}, sk_{S_1}) and V's key pair (pk_V, sk_V). Then, $\mathcal{B}$ sends $(sp, pk_{S_0}, pk_{S_1}, pk_V)$ to $\mathcal{D}$.
- **Phase 1**: $\mathcal{D}$ may adaptively issue the following oracle queries.
  - $\mathcal{O}_{Sign}$: Upon receiving sp, a message m_i and an index $d \in \{0,1\}$, this oracle returns a signature $\sigma_i = \textbf{Sign}(sk_{S_d}, pk_V, m_i)$ of m_i.
  - $\mathcal{O}_{Sim}$: Upon receiving sp, a message m_i and an index $d \in \{0,1\}$, this oracle returns a simulated signature $\sigma_i^{'} = \textbf{Sim}(sk_V, pk_{S_d}, m_i)$ on m_i.
  - $\mathcal{O}_{Verify}$: Taking sp, an index $d \in \{0,1\}$ and a signature σ on a message m as input, this oracle outputs a decision $dec = \textbf{Verify}(sk_V, pk_{S_d}, m, \sigma)$, where $dec \in \{0,1\}$.
  - $\mathcal{O}_{ReSign}$: Upon input of sp, a message m_d, an index $d \in \{0,1\}$, a signature σ_d on m_d and an index $d_2 \in \{0,1\}$, this oracle outputs a re-signature $\sigma_{d_2} = \textbf{ReSign}(\textbf{ReVKey}(pk_d, sk_d, pk_{d_2}, sk_{d_2}), m_d, pk_d, \sigma_d)$ on m_d.
- **Challenge**: Upon receiving a challenge message m^* submitted by $\mathcal{D}$, $\mathcal{B}$ first selects a random bit $b \in \{0,1\}$ and runs the algorithm **Sign** to generate a signature $\sigma^* = \textbf{Sign}(sk_{S_b}, pk_V, m^*)$. Then, $\mathcal{B}$ returns σ^* to $\mathcal{D}$.
- **Phase 2**: $\mathcal{D}$ continues to query $\mathcal{O}_{Sign}$, $\mathcal{O}_{Sim}$ and $\mathcal{O}_{Verify}$ defined in Phase 1, but $\mathcal{D}$ is not allowed to submit (m^*, σ^*) to $\mathcal{O}_{Verify}$.
- **Guessing Phase**: $\mathcal{D}$ finally outputs a guess $b^{'} \in \{0,1\}$. The distinguisher $\mathcal{D}$ wins the game if $b = b^{'}$.

The advantage of $\mathcal{D}$ in the game is defined as follows:

$$Adv_D = \left| \Pr[b^{'} = b] - \frac{1}{2} \right|.$$

Definition 6. *If the advantage Adv_D of any polynomial-time distinguisher $\mathcal{D}$ is negligible in the above game, then we say that an SDVPRS scheme possesses the PSI property.*

Non-delegatability for signature verification requires that the legality of a signature can only be correctly verified by those who know the secret key of the designated verifier. We use a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ to define non-delegatability for signature verification.

- **Setup**: $\mathcal{C}$ executes the algorithms **Setup** and **KeyGen** to generate system parameters sp, the key pair (pk_S, sk_S) of the target signer and the key pair (pk_V, sk_V) of the target verifier. Then, $\mathcal{C}$ sends (sp, pk_S, pk_V) to $\mathcal{A}$.

- **Queries**: $\mathcal{C}$ answers $\mathcal{A}$'s signing query and simulation query in the same manner as in the formal definition of existential unforgeability in Definition 5.
- **Forgery**: $\mathcal{A}$ finally produces a forgery (m^*, σ^*). If the following conditions are satisfied, then we say that $\mathcal{A}$ wins the game.

1. σ^* is a valid signature on m^*, namely **Verify**$(sk_V, pk_S, m^*, \sigma^*) = 1$.
2. $\mathcal{A}$ has never made a signature query and a simulated query on m^*.

Definition 7. *An SDVPRS scheme is said to be non-delegatable for signature verification if there is no polynomial-time attacker $\mathcal{A}$ who can win the above game with a non-negligible probability.*

3.3. System Framework

Our system model is shown in Figure 1, which contains three entities: IoT device, proxy and data center. It focuses on the integrity and authenticity of IoT data during transmission while protecting the identity privacy of the IoT device. The proposed SDVPRS scheme is easily implemented on each IoT device as software. Our system model focuses on the integrity and authenticity of IoT data.

- IoT device: This entity has very limited computing and communication capabilities. Each IoT device uses its secret key to sign the data collected from the physical world and then sends the data and its signature to the proxy.
- Proxy: This entity is usually served by a semi-trusted server with a certain computation and communication power. To ensure the security of data storage, IoT data are stored by multiple data centers for consumers of different security levels. The proxy uses the re-signing key $rsk_{A\to B}$ to convert the signature σ_{AC} generated by the IoT device into a group's signature σ_{BC} for the same message m, so that the data center only knows that σ_{BC} is a valid signature on m, but cannot infer the identity of the real signer. After receiving (m, σ_{BC}), another proxy uses the re-designate-verifier key $rvk_{C\to D_i}$ to convert the signature σ_{BC} into the signature σ_{BD_i} for every data center, and it sends the IoT data m and σ_{BD_i} to the i-th data center, where $i \in \{1, ..., n\}$.
- Data center: This entity has high computing and storage capacities. After verifying the validity of the received signature, the authentic IoT data are stored by each data center.

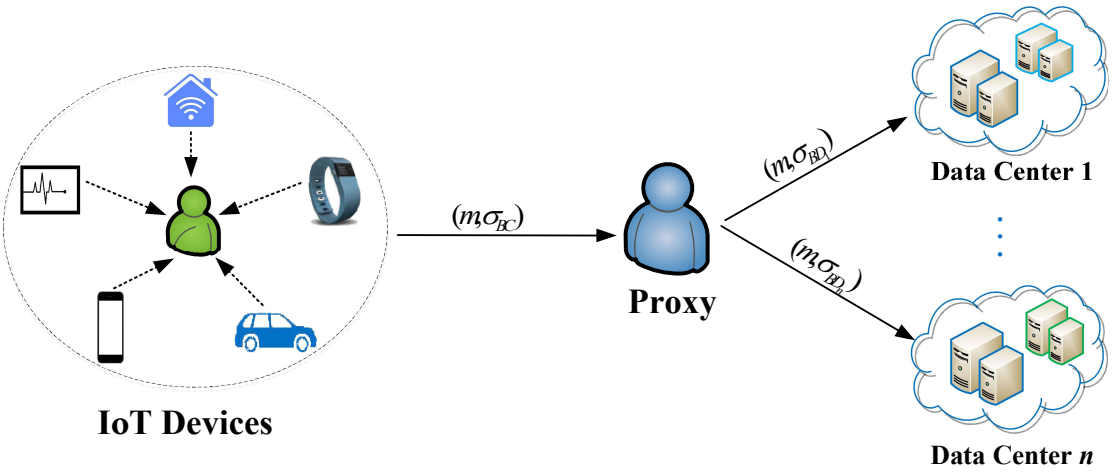

Figure 1. System framework for IoT data authenticity.

4. Our SDVPRS Scheme

In this section, we present a construction of an SDVPRS scheme based on Waters' scheme [19]. In our proposed SDVPRS scheme, the length of a message is assumed to be n bits. For a message of

arbitrary length, we use a hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$ to convert the length of the message to fixed length n. Our SDVPRS scheme works as follows.

- **Setup**: This algorithm takes a security parameter $\lambda \in Z$ as input and produces system parameters $sp = (G_1, G_2, p, g, e, u_0, u_1, ..., u_n)$, where G_1 and G_2 are two cyclic groups of prime order p, g is a generator of G_1, $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear pairing and $n+1$ elements $u_0, u_1, ..., u_n$ are randomly chosen from G_1.
- **KeyGen**: The signer S randomly selects $(x_S, y_S) \in Z_p$, computes $pk_{S,1} = g^{x_S}$ and $pk_{S,2} = g^{y_S}$ and sets its secret key $sk_S = (sk_{S,1}, sk_{S,2}) = (x_S, y_S)$ and the public key $pk_S = (pk_{S,1}, pk_{S,2}) = (g^{x_S}, g^{y_S})$. Similarly, the designated verifier V randomly selects $x_V \in Z_p$, computes $pk_V = g^{x_V}$ and outputs its public/secret key pair $(pk_V, sk_V) = (g^{x_V}, x_V)$.
- **ReSKey**: The proxy randomly selects $r_1 \in Z_p$ and sends r_1 to the signer S_A. After receiving r_1, S_A uses his/her secret key $sk_A = (x_A, y_A)$ to compute $r_2 = x_A y_A r_1 (\text{mod } p)$ and forwards r_2 to the signer S_B. Then, S_B uses his/her secret key $sk_B = (x_B, y_B)$ to calculate and send $r_3 = \frac{x_B y_B}{r_2} (\text{mod } p)$ to the proxy. Finally, the proxy computes the re-signing key between S_A and S_B:

$$rsk_{A \rightarrow B} = r_3 r_1 = r_1 (\frac{x_B y_B}{x_A y_A r_1}) = \frac{x_B y_B}{x_A y_A} (\text{mod } p).$$

- **ReVKey**: To generate a re-designate-verifier key $rvk_{C \rightarrow D}$ between two verifiers V_C and V_D, the proxy does the following:

  1. The proxy randomly selects $s_1 \in Z_p$ and returns s_1 to the verifier V_C.
  2. V_C uses his/her secret key $sk_C = x_C$ to compute $s_2 = s_1 x_C (\text{mod } p)$ and sends s_2 to the verifier V_D.
  3. V_D uses his/her secret key $sk_D = x_D$ to compute $s_3 = \frac{x_D}{s_2} (\text{mod } p)$. Then, V_D returns s_3 to the proxy.
  4. After receiving s_3, the proxy calculates the re-designate-verifier key as follows:

$$rvk_{C \rightarrow D} = s_3 s_1 = s_1 (\frac{x_D}{s_1 x_C}) = \frac{x_D}{x_C} (\text{mod } p).$$

  Note that $rsk_{A \rightarrow B}$ and $rvk_{C \rightarrow D}$ can be kept by different proxies to perform different operations, and they can also be assigned to a proxy to perform all operations. In **ReSKey** and **ReVKey**, a cryptographic algorithm (such as RSA or ECC) may be used to encrypt the transmitted messages to prevent the proxy from obtaining each participant's secret key through intercepted messages in practical applications.
- **Sign**: Given a message $m = (m_1, ...m_n) \in \{0,1\}^n$ and a designated verifier V's public key g^v, the signer S performs the following steps:

  1. Choose a random integer $r \in Z_p$, and calculate $\sigma_2 = g^r$.
  2. Calculate $\sigma_1 = e(g^{x_S y_S} (u_0 \prod_{j=1}^{n} u_i^{m_i})^r, g^v)$, where (x_S, y_S) is the secret key of the signer S.
  3. Output $\sigma = (\sigma_1, \sigma_2)$ as a signature on m.
- **ReSign**: For a signature $\sigma_{AC} = (\sigma_{AC,1}, \sigma_{AC,2})$ on a message m with respect to a signer S_A and a verifier V_C, the proxy uses a re-signing key $rsk_{A \rightarrow B}$ between two signers S_A and S_B to compute a new signature σ_{BC} on m related to the signer S_B and the verifier V_C as follows:

$$\sigma_{BC} = ((\sigma_{AC,1})^{rsk_{A \rightarrow B}}, (\sigma_{AC,2})^{rsk_{A \rightarrow B}}).$$

- **ReVer**: Given a re-designate-verifier key $rvk_{C \rightarrow D}$ between two verifiers V_C and V_D, a message m and a signature $\sigma_{AC} = (\sigma_{AC,1}, \sigma_{AC,2})$ of m for a signer S_A and a verifier V_C, the proxy computes a new signature σ_{AD} on m for S_A and V_D as follows:

$$\sigma_{AD} = ((\sigma_{AC,1})^{rvk_{C \rightarrow D}}, (\sigma_{AC,2})^{rvk_{C \rightarrow D}}).$$

- **Verify**: Upon receiving the public key $pk_S = (g^{x_S}, g^{y_S})$ of a signer S, a message m and a corresponding signature $\sigma = (\sigma_1, \sigma_2) = (e(g^{x_S y_S}(u_0 \prod_{i=1}^{n} u_i^{m_i})^r, g^{x_V}), g^r)$, the designated verifier V utilizes its own secret key $sk_V = x_V$ to check whether:

$$\sigma_1 = e(g^{x_S}, g^{y_S})^{x_V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_2)^{x_V}$$

holds. If it holds, output one; else, output zero.
- **Sim**: For a message m and the public key $pk_S = (g^{x_S}, g^{y_S})$ of a signer S, the designated verifier V with a secret key $sk_V = x_V$ randomly selects $r^{'} \in Z_p$, computes $\sigma_2^{'} = g^{r^{'}}$ and:

$$\sigma_1^{'} = e(g^{x_S}, g^{y_S})^{x_V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_2^{'})^{x_V},$$

then generates $\sigma^{'} = (\sigma_1^{'}, \sigma_2^{'})$ as a simulated signature on m with respect to S and V.

Correctness: Assume that $\sigma_{AC} = (\sigma_{AC,1}, \sigma_{AC,2}) = (e(g^{x_A y_A}(u_0 \prod_{i=1}^{n} u_i^{m_i})^{r_A}, g^{x_V}), g^{r_A})$ is a signature on a message m for S_A and V_C, $rsk_{A\to B} = \frac{x_B y_B}{x_A y_A} (\mathrm{mod}\ p)$ is a re-signing key between S_A and S_B and $\sigma_{BC} = (\sigma_{BC,1}, \sigma_{BC,2})$ is a signature on m that is derived from σ_{AC} and $rsk_{A\to B}$.

Let $\hat{r}_A = r_A$; we have:

$$\begin{aligned}
\sigma_{BC,1} &= (\sigma_{AC,1})^{rsk_{A\to B}} \\
&= e(g^{x_A y_A}(u_0 \prod_{i=1}^{n} u_i^{m_i})^{r_A}, g^{x_V})^{\frac{x_B y_B}{x_A y_A}} \\
&= e(g^{(x_A y_A)\frac{x_B y_B}{x_A y_A}}(u_0 \prod_{i=1}^{n} u_i^{m_i})^{r_A \frac{x_B y_B}{x_A y_A}}, g^{x_V}) \\
&= e(g^{x_B y_B}(u_0 \prod_{i=1}^{n} u_i^{m_i})^{\hat{r}_A}, g^{x_V}),
\end{aligned}$$

$$\sigma_{BC,2} = (\sigma_{AC,2})^{rsk_{A\to B}} = (g^{r_A})^{\frac{x_B y_B}{x_A y_A}} = g^{r_A \frac{x_B y_B}{x_A y_A}} = g^{\hat{r}_A}.$$

Then, we can obtain:

$$\begin{aligned}
\sigma_{BC,1} &= e(g^{x_B y_B}(u_0 \prod_{i=1}^{n} u_i^{m_i})^{\hat{r}_A}, g^{x_V}) \\
&= e(g^{x_B y_B}, g^{x_V}) e((u_0 \prod_{i=1}^{n} u_i^{m_i})^{\hat{r}_A}, g^{x_V}) \\
&= e(g^{x_B}, g^{y_B})^{x_V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, g^{\hat{r}_A})^{x_V} \\
&= e(g^{x_B}, g^{y_B})^{x_V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_{BC,2})^{x_V}.
\end{aligned}$$

Similarly, we are able to check the correctness of the signature σ_{AD} for S_A and V_D. Hence, we can conclude that our SDVPRS scheme is correct.

Remark 1. *For a re-signing key $rsk_{A\to B}$ between S_A and S_B, it is easy to obtain another key $rsk_{B\to A} = \frac{1}{rsk_{A\to B}}$ between S_B and S_A that converts S_B's signatures into S_A's signatures. From a re-designate-verifier key $rvk_{C\to D}$ between V_C and V_D, we can also obtain another key $rvk_{D\to C} = \frac{1}{rvk_{C\to D}}$ that transforms the identity of the verifier in a signature from V_D to V_C. Hence, our SDVPRS scheme is bidirectional.*

Remark 2. *As* $\sigma_{BC} = (\sigma_{AC})^{rsk_{A\to B}}$ *and* $\sigma_{AD} = (\sigma_{AC})^{rvk_{C\to D}}$, *it is easy to infer that signatures created by the* **Sign** *algorithm are computationally indistinguishable from signatures generated by the* **ReSign** *and* **ReVer** *algorithms. This shows that the proposed SDVPRS scheme possesses the multi-use property.*

Remark 3. *If* $\sigma = (e(g^{x_S y_S}(u_0 \prod_{i=1}^{n} u_i^{m_i})^r, g^{x_V}), g^r)$ *is a signature created by the signer S for the message m, then the designated verifier V can also output a valid signature* $\sigma' = (e(g^{x_S}, g^{y_S})^{x_V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, g^r)^{x_V}, g^r)$ *on the same message m. Consequently, the distribution of* σ *is the same as that of* σ'. *This result implies that signatures generated by the* **Sign** *algorithm are indistinguishable from those simulated by the* **Sim** *algorithm. Therefore, our SDVPRS scheme is non-transferable.*

Theorem 1. *If the GBDH problem is intractable, then our proposed SDVPRS scheme is existentially unforgeable against adaptively chosen message attacks in the standard model.*

Proof of Theorem 1. Let $\mathcal{A}$ be an attacker against the unforgeability of the proposed scheme with probability ε. $\mathcal{A}$ is allowed to make at most q_S signing queries, q_V signature verification queries, q_{Sim} simulation queries, q_{rsk} re-signing key queries and q_{rvk} re-designate-verifier key queries. We can build an algorithm $\mathcal{C}$ that utilizes $\mathcal{A}$'s output to solve the GBDH problem. Given an instance $(g, g^a, g^b, g^c) \in G_1^4$ of the GBDH problem, $\mathcal{C}$'s goal is to calculate $e(g,g)^{abc}$ by invoking the oracle $\mathcal{O}_{DBDH}$. $\mathcal{C}$ will act as a challenger to answer the following queries requested by $\mathcal{A}$.

- **Setup**: $\mathcal{C}$ sets $l_m = 2(q_S + q_V + q_{Sim})$ such that $l_m(n+1) < p$. Then, $\mathcal{C}$ chooses a random integer $k_m (0 \le k_m \le n)$, $n+1$ random values $v_0, v_1, ..., v_n \in Z_{l_m}$ and $n+1$ random elements $w_0, w_1, ..., w_n \in Z_p$. Furthermore, $\mathcal{C}$ assigns $u_0 = (g^b)^{v_0 - l_m k_m} g^{w_0}$ and $u_i = (g^b)^{v_i} g^{w_i}$ for $i = 1, ..., n$. In addition, $\mathcal{C}$ randomly selects $x_i, y_i, z_j \in Z_p$ and sets the public key of the signer i to be $pk_i = (g^{ax_i}, g^{by_i})$ and the public key of the verifier j as $pk_j = g^{cz_j}$ for $i = 1, ..., 2q_{rsk}$ and $j = 1, ..., 2q_{rvk}$. The public key of the target signer is set to $pk_S = (pk_{S,1}, pk_{S,2}) = (g^a, g^b)$, and the public key of the target designated verifier's public key is set to $pk_V = g^c$. This implicitly indicates that $sk_S = (a, b)$ is the target signer's secret key and $sk_V = c$ is the target designated verifier's secret key, but (a, b) and c are unknown to $\mathcal{C}$. Finally, $\mathcal{C}$ sends $(G_1, G_2, p, g, e, u_0, u_1, ..., u_n, pk_S, pk_V)$ and the public keys of other users to $\mathcal{A}$.

  Given any n-bit message $m = (m_1, ..., m_n)$, we define two functions to simplify the expression: $F(m) = v_0 - l_m k_m + \sum_{i=1}^{n} v_i m_i$ and $J(m) = w_0 + \sum_{i=1}^{n} w_i m_i$. Consequently, we have $u_0 \prod_{i=1}^{n} u_i^{m_i} = (g^b)^{F(m)} g^{J(m)}$.
- **Queries**: $\mathcal{C}$ builds the following oracles to answer $\mathcal{A}$'s queries.
  - $\mathcal{O}_{Sign}$: Upon receiving a message m, $\mathcal{C}$ checks whether $F(m) = 0 (\mathrm{mod}\ l_m)$ holds. If it does hold, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ randomly selects $r \in Z_p$, computes $\sigma_1 = e((g^a)^{\frac{-J(m)}{F(m)}} (u_0 \prod_{i=1}^{n} u_i^{m_i})^r, g^c)$ and $\sigma_2 = (g^a)^{\frac{-1}{F(m)}} g^r$ and then returns a signature $\sigma = (\sigma_1, \sigma_2)$ on m to $\mathcal{A}$.

Correctness: For a signature $\sigma = (\sigma_1, \sigma_2)$ produced by $\mathcal{C}$, we have:

$$\begin{aligned}
\sigma_1 &= e((g^a)^{\frac{-J(m)}{F(m)}}(u_0\prod_{i=1}^{n}u_i^{m_i})^r, g^c) \\
&= e((g^{ab})(g^{-ab})(g^a)^{\frac{-J(m)}{F(m)}}(u_0\prod_{i=1}^{n}u_i^{m_i})^r, g^c) \\
&= e(g^{ab}((g^b)^{F(m)}(g^{J(m)}))^{\frac{-a}{F(m)}}(u_0\prod_{i=1}^{n}u_i^{m_i})^r, g^c) \\
&= e(g^{ab}(u_0\prod_{i=1}^{n}u_i^{m_i})^{\frac{-a}{F(m)}}(u_0\prod_{i=1}^{n}u_i^{m_i})^r, g^c) \\
&= e(g^{ab}(u_0\prod_{i=1}^{n}u_i^{m_i})^{r-\frac{a}{F(m)}}, g^c) \\
&= e(g^{ab}(u_0\prod_{i=1}^{n}u_i^{m_i})^{\hat{r}}, g^c)
\end{aligned}$$

and $\sigma_2 = (g^a)^{\frac{-1}{F(m)}}g^r = g^{r-\frac{a}{F(m)}} = g^{\hat{r}}$, where $\hat{r} = r - \frac{a}{F(m)}$.
Then, we deduce that $\sigma = (\sigma_1, \sigma_2)$ satisfies the following signature verification equation:

$$\begin{aligned}
\sigma_1 &= e(g^{ab}(u_0\prod_{i=1}^{n}u_i^{m_i})^{\hat{r}}, g^c) \\
&= e(g^{ab}, g^c)e((u_0\prod_{i=1}^{n}u_i^{m_i})^{\hat{r}}, g^c) \\
&= e(g^a, g^b)^c e(u_0\prod_{i=1}^{n}u_i^{m_i}, \sigma_2)^c.
\end{aligned}$$

This equation shows that the signature σ is valid. Moreover, it indicates that the signature computed by $\mathcal{C}$ and the signature created by the signer in the real scheme are computationally indistinguishable from the adversary $\mathcal{A}$'s view.

- $\mathcal{O}_{ReSKey}$: Upon receiving two signers' public keys pk_i and pk_j, $\mathcal{C}$ computes $rsk_{i\to j} = \frac{ax_j \cdot by_j}{ax_i \cdot by_i} = \frac{x_j \cdot y_j}{x_i \cdot y_i} (\mathrm{mod}\, p)$ and returns a re-signing key $rsk_{i\to j}$ to $\mathcal{A}$.
- $\mathcal{O}_{ReVKey}$: Upon input of two verifiers' public keys pk_i and pk_j, $\mathcal{C}$ computes $rvk_{i\to j} = \frac{cz_j}{cz_i} = \frac{z_j}{z_i} (\mathrm{mod}\, p)$ and returns a corresponding re-designate-verifier key $rvk_{i\to j}$ to $\mathcal{A}$.
- $\mathcal{O}_{Sim}$: This oracle is the same as $\mathcal{O}_{Sign}$.
- $\mathcal{O}_{Verify}$: Upon receiving a message m and a signature $\sigma = (\sigma_1, \sigma_2)$, $\mathcal{C}$ responds by performing the following steps:

  (1) If $F(m) \neq 0 (\mathrm{mod}\ p)$, then $\mathcal{C}$ creates another signature $\tilde{\sigma} = (\tilde{\sigma}_1, \tilde{\sigma}_2)$ of m just as $\mathcal{C}$ responds to the signing queries in the oracle $\mathcal{O}_{Sign}$. Subsequently, $\mathcal{C}$ submits $(g, u_0\prod_{i=1}^{n}u_i^{m_i}, \frac{\sigma_2}{\tilde{\sigma}_2}, g^c, \frac{\sigma_1}{\tilde{\sigma}_1})$ to the oracle $\mathcal{O}_{DBDH}$. If $\mathcal{O}_{DBDH}$ outputs one, then $\mathcal{C}$ sends one to $\mathcal{A}$; otherwise, $\mathcal{C}$ sends zero to $\mathcal{A}$.
  Correctness: If $\sigma = (\sigma_1, \sigma_2)$ is a valid signature for message m, then σ must satisfy the following signature verification equation:

$$\sigma_1 = e(g^a, g^b)^c e(u_0\prod_{i=1}^{n}u_i^{m_i}, \sigma_2)^c.$$

Another valid signature $\tilde{\sigma} = (\tilde{\sigma}_1, \tilde{\sigma}_2)$ calculated by $\mathcal{C}$ must also satisfy the following verification equation:

$$\tilde{\sigma}_1 = e(g^a, g^b)^c e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \tilde{\sigma}_2)^c.$$

Thus, we can obtain:

$$\begin{aligned}\frac{\sigma_1}{\tilde{\sigma}_1} &= \frac{e(g^a, g^b)^c e(u_0 \prod\limits_{i=1}^{n} u_i^{m_i}, \sigma_2)^c}{e(g^a, g^b)^c e(u_0 \prod\limits_{i=1}^{n} u_i^{m_i}, \tilde{\sigma}_2)^c} \\ &= e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \frac{\sigma_2}{\tilde{\sigma}_2})^c.\end{aligned}$$

The above equation demonstrates that $(g, u_0 \prod\limits_{i=1}^{n} u_i^{m_i}, \frac{\sigma_2}{\tilde{\sigma}_2}, g^c, \frac{\sigma_1}{\tilde{\sigma}_1})$ is a correct BDH tuple.

(2) If $F(m) = 0(\text{mod } p)$, $\mathcal{C}$ makes a query to the oracle $\mathcal{O}_{DBDH}$ with input $(g, g^a, g^b, g^c, \frac{\sigma_1}{e(g^c, \sigma_2)^{J(m)}})$. If $\mathcal{O}_{DBDH}$ returns one, then $\mathcal{C}$ sends one to $\mathcal{A}$; else, $\mathcal{C}$ sends zero to $\mathcal{A}$.
Correctness: Assuming that $\sigma = (\sigma_1, \sigma_2)$ is a valid signature, the following equation holds:

$$\begin{aligned}\sigma_1 &= e(g^a, g^b)^c e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_2)^c \\ &= e(g^a, g^b)^c e((g^b)^{F(m)} g^{J(m)}, \sigma_2)^c \\ &= e(g^a, g^b)^c e(g^{J(m)}, \sigma_2)^c \\ &= e(g^a, g^b)^c e(g^c, \sigma_2)^{J(m)}.\end{aligned}$$

Hence, we can obtain:

$$e(g, g)^{abc} = \frac{\sigma_1}{e(g^c, \sigma_2)^{J(m)}}.$$

This shows that $(g, g^a, g^b, g^c, \frac{\sigma_1}{e(g^c, \sigma_2)^{J(m)}})$ is a valid BDH tuple.

- **Forgery**: $\mathcal{A}$ produces a message/signature pair $(m^*, \sigma^*) = (m^*, (\sigma_1^*, \sigma_2^*))$, where m^* has never been submitted to $\mathcal{O}_{Sign}$ and $\mathcal{O}_{Sim}$. If $F(m^*) \neq 0(\text{mod } p)$, then $\mathcal{C}$ terminates the simulation. Otherwise, $\mathcal{C}$ can successfully obtain a solution for the given CDHinstance by calculating:

$$e(g, g)^{abc} = \frac{\sigma_1^*}{e(g^c, \sigma_2^*)^{J(m^*)}}.$$

We now analyze the probability that $\mathcal{C}$ can successfully solve the GBDH instance. If $\mathcal{C}$ completes the entire simulation without aborting, then the following events must occur.

- A_1**:** All signature, simulation and verification queries on any message m_j satisfy $F(m_j) \neq 0(\text{mod } l_m)$.
- A_2**:** $F(m^*) = 0(\text{mod } p)$.

As $F(m) = v_0 - l_m k_m + \sum\limits_{i=1}^{n} v_i m_i$ and $n+1$ random elements $v_0, v_1, ..., v_n \in Z_{l_m} \in Z_p$, $F(m) = 0(\text{mod } p)$ implies that $F(m) = 0(\text{mod } l_m)$. In contrast, $F(m) \neq 0(\text{mod } l_m)$ implies that $F(m) \neq 0(\text{mod } p)$. The probability of $\mathcal{C}$ completing the entire simulation is $\Pr[A_1 \cap A_2]$. Since $\Pr[F(m^*) = 0(\text{mod } l_m)] = \frac{1}{l_m}$ and $l_m(n+1) < p$, we have:

$$\begin{aligned}\Pr[A_2] &= \Pr[F(m^*) = 0(\text{mod } l_m) \cap F(m^*) = 0(\text{mod } p)] \\ &\geq \frac{1}{l_m} \frac{1}{n+1}.\end{aligned}$$

Since two events A_1 and A_2 are independent, we have:

$$\begin{aligned}\Pr[A_1|A_2] &= \Pr[A_1] \\ &= \Pr[\bigcap_{j=1}^{q_S+q_{Sim}+q_V} F(m_j) \neq 0 (\text{mod } l_m)] \\ &= 1 - \frac{q_S + q_{Sim} + q_V}{l_m} \\ &= 1 - \frac{q_S + q_{Sim} + q_V}{2(q_S + q_{Sim} + q_V)} \\ &= \frac{1}{2}.\end{aligned}$$

Furthermore, we conclude that:

$$\begin{aligned}\Pr[A_1 \cap A_2] &= \Pr[A_2] \cdot \Pr[A_1|A_2] \\ &\geq \frac{1}{l_m} \cdot \frac{1}{n+1} \cdot \frac{1}{2} \\ &= \frac{1}{2(q_S + q_{Sim} + q_V)} \cdot \frac{1}{2(n+1)} \\ &= \frac{1}{4(n+1)(q_S + q_{Sim} + q_V)}.\end{aligned}$$

The probability ε' of $\mathcal{C}$ successfully solving the GBDH problem is at least $\frac{\varepsilon}{4(n+1)(q_S+q_{Sim}+q_V)}$. □

Theorem 2. *If the DBDH problem is intractable, then our SDVPRS scheme possesses the PSI property.*

Proof of Theorem 2. Let $\mathcal{D}$ be a polynomial-time distinguisher against the PSI property in the proposed SDVPRS scheme. We can construct another algorithm $\mathcal{B}$ that invokes $\mathcal{D}$ to solve the DBDH problem. Upon receiving a DBDH instance (g, g^a, g^b, g^c, Z), where $a, b, c \in Z_p$ and $Z \in G_2$, the task of $\mathcal{B}$ is to determine whether $Z = e(g,g)^{abc}$ holds.

- **Setup**: $\mathcal{B}$ selects $x, y, w_0, w_1, ..., w_n \in Z_p$ at random and sets $u_0 = g^{w_0}$ and $u_i = g^{w_i} (1 \leq i \leq n)$. Then, $\mathcal{B}$ assigns the signer S_0's public key $pk_{S_0} = (g^a, g^b)$, the signer S_1's public key $pk_{S_1} = (g^x, g^y)$ and the public key $pk_V = g^c$ of the designated verifier V. $\mathcal{B}$ sets the common secret key between S_0 and V as $sk_{S_0 \leftrightarrow V} = Z$. Additionally, $\mathcal{B}$ sets the common secret key between S_1 and V as $sk_{S_1 \leftrightarrow V} = e(g^x, g^c)^y$. Finally, $\mathcal{B}$ sends $(G_1, G_2, p, g, e, u_0, u_1, ..., u_n, pk_{S_0}, pk_{S_1}, pk_V)$ to $\mathcal{D}$.

  Given any n-bit message $m = (m_1, ..., m_n)$, we define a function $K(m) = w_0 + \sum_{i=1}^{n} w_i m_i$. Thus, we have:

$$u_0 \prod_{i=1}^{n} u_i^{m_i} = g^{K(m)}.$$

- **Phase 1**: $\mathcal{D}$ adaptively issues queries to the following oracles.
  - $\mathcal{O}_{Sign}$: Upon receiving an index $d \in \{0,1\}$ and a message m, $\mathcal{B}$ selects $r \in Z_p$ at random and uses the common secret key $sk_{S_d \leftrightarrow V}$ to compute:

$$\sigma_1 = sk_{S_d \leftrightarrow V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, g^c)^r$$

    and $\sigma_2 = g^r$. Then, $\mathcal{B}$ sends a signature $\sigma = (\sigma_1, \sigma_2)$ of m to $\mathcal{D}$.
  - $\mathcal{O}_{Sim}$: This oracle is the same as $\mathcal{O}_{Sign}$.

– $\mathcal{O}_{Verify}$: When $\mathcal{D}$ issues a verification query on a signature $\sigma = (\sigma_1, \sigma_2)$ of a message m and an index $d \in \{0,1\}$, $\mathcal{B}$ computes $K(m)$ and uses $sk_{S_d \leftrightarrow V}$ to check whether the following equation is true:

$$\sigma_1 = sk_{S_d \leftrightarrow V} e(\sigma_2, g^c)^{K(m)}.$$

If this equation holds, then $\mathcal{B}$ sends one to $\mathcal{D}$; else, $\mathcal{B}$ sends zero to $\mathcal{D}$.

Correctness: For a signature $\sigma = (\sigma_1, \sigma_2)$ associated with the signer S_d and the verifier V, we have:

$$\begin{aligned}\sigma_1 &= sk_{S_d \leftrightarrow V} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, g^c)^r \\ &= sk_{S_d \leftrightarrow V} e(g^{K(m)}, g^c)^r \\ &= sk_{S_d \leftrightarrow V} e(g^r, g^c)^{K(m)} \\ &= sk_{S_d \leftrightarrow V} e(\sigma_2, g^c)^{K(m)}.\end{aligned}$$

This shows that $\mathcal{B}$ can correctly verify the legality of the signature submitted by $\mathcal{D}$.

– $\mathcal{O}_{ReSign}$: Upon input of a message m_d, an index $d \in \{0,1\}$, a signature σ_d on m_d and an index $d_2 \in \{0,1\}$, $\mathcal{B}$ first performs a verification query $\mathcal{O}_{Verify}$ with input (m_d, d, σ_d) to obtain a corresponding decision $dec \in \{0,1\}$. If $dec = 0$, then $\mathcal{B}$ returns $\perp$; otherwise, $\mathcal{B}$ obtains a signature σ_{d_2} related to S_{d_2} and V by querying the oracle $\mathcal{O}_{Sign}$ with input (m_d, d_2). Then, $\mathcal{B}$ sends σ_{d_2} to $\mathcal{D}$ as a re-signature on m_d.

- **Challenge**: After receiving a challenge message m^* submitted by $\mathcal{D}$, $\mathcal{B}$ first randomly selects a bit $b \in \{0,1\}$ and queries the oracle $\mathcal{O}_{Sign}$ on input (m^*, b) to obtain a corresponding signature σ^* of m^*. Then, $\mathcal{B}$ sends σ^* to $\mathcal{D}$.

  Note that if $b = 0$, then $Z = e(g,g)^{abc}$; otherwise, $Z = e(g^x, g^c)^y$ is a random element in G_2.
- **Phase 2**: $\mathcal{B}$ handles the subsequent queries submitted by $\mathcal{D}$ as in Phase 1, and $\mathcal{D}$ is not allowed to query $\mathcal{O}_{Verify}$ with input (m^*, σ^*, d), where $d \in \{0,1\}$.
- **Guessing Phase**: $\mathcal{D}$ eventually produces a guess $b' \in \{0,1\}$. $\mathcal{B}$ outputs one if $b = b'$ and zero otherwise.

If $\mathcal{D}$ can correctly guess b' such that $b = b'$, then we have:

$$\begin{aligned}\varepsilon =& |\Pr[b = b' \cap z = e(g,g)^{abc}] - \Pr[b = b' \cap z = \\ & e(g^x, g^c)^y]| \\ \leq & \frac{1}{2}(|\Pr[b = b' | z = e(g,g)^{abc}] - \Pr[b = b' | z = \\ & e(g^x, g^c)^y]|) \\ = & \frac{1}{2}\varepsilon_{DBDH}.\end{aligned}$$

Therefore, $\mathcal{B}$ can successfully solve a DBDH instance with at least 2ε probability. □

Theorem 3. *If the BDH problem is intractable, then our SDVPRS scheme is non-delegatable for signature verification.*

Proof of Theorem 3. Let $\mathcal{A}$ be a polynomial-time attacker against the property of non-delegatability for signature verification. Given a BDH tuple $(g, g^a, g^b, g^c) \in G_1^4$, we can construct another algorithm $\mathcal{C}$ that utilizes $\mathcal{A}$'s forgery to calculate $Z = e(g,g)^{abc}$.

- **Setup**: $\mathcal{C}$ sets system parameters sp, the signer's public key $pk_S = (g^a, g^b)$ and the designated verifier's public key $pk_V = g^c$ in the same way as in the proof of Theorem 1.
- **Queries**: $\mathcal{C}$ answers $\mathcal{A}$'s signing and simulation queries in the same manner as in the proof of Theorem 1.

- **Forgery**: Eventually, $\mathcal{A}$ produces a message/signature pair $(m^*, \sigma^* = (\sigma_1^*, \sigma_2^*))$. If $F(m^*) \neq 0 (\text{mod } p)$, then $\mathcal{C}$ terminates the simulation. Otherwise, $\mathcal{C}$ evaluates $J(m^*)$ and outputs the BDH value:

$$e(g,g)^{abc} = \frac{\sigma_1^*}{e(g^c, \sigma_2^*)^{J(m^*)}}.$$

The probabilistic analysis of $\mathcal{C}$ successfully solving the BDH problem is similar to the probability analysis of Theorem 1. □

5. Performance Evaluation

This section discusses the performance comparison between our SDVPRS scheme and Wei et al.'s [14] DVPRS scheme with respect to the security model, the size of the secret key, the signature length and the computational overhead. The corresponding comparison results are shown in Tables 1 and 2. We use the PBClibrary to evaluate the time cost of cryptographic operations. We select the curve *a.param* of Type A in the PBC-0.47-VC library to perform bilinear pairing, and p is a 512-bit prime. The simulated environment is set up on a laptop with the Windows 10 operating system, with an Intel(R) Core(TM) i7-6500 CPU @2.59 GHz and 8 GB of RAM. Since the computational overhead of other cryptographic operations is relatively small, we mainly consider time-consuming exponentiation and bilinear pairing. To simplify the description, let E denote an exponentiation computation that takes 7.46 ms. P represents a bilinear pairing operation, which takes 14.13 ms. $|p|$ represents the length of an element in Z_p, which is 20 bits. $|G_1|$ represents the length of an element in G_1, which is 128 bits. In Table 1, *KeySize* and *SigSize* are used to represent the size of a secret key and a signature, respectively.

Table 1. Comparison of communication overhead. SigSize, signature size; SDVPRS, strong designated-verifier proxy re-signature.

Schemes	KeySize	SigSize	Standard Model
Wei et al.'s scheme [14]	$2\|p\|$	$\|G_1\|$	No
Our SDVPRS scheme	$2\|p\|$	$2\|G_1\|$	Yes

Table 2. Comparison of computational overhead.

Schemes	Sign	ReSign	ReVer	Verify
Wei et al.'s scheme [14]	$E+P$	E	E	$E+2P$
Our SDVPRS scheme	$2E+P$	$2E$	$2E$	$E+P$

Table 1 presents the comparisons of the communication overhead based on the size of the secret key and signature. In both our SDVPRS scheme and Wei et al.'s scheme [14], the size of the secret key is $2|p|$ (40 bits). In our SDVPRS scheme, the length of a signature is $2|G_1|$ (256 bits). The length of a signature in Wei et al.'s scheme [14] is $|G_1|$ (128 bits), but this scheme has been proven to be secure in the random oracle model.

Table 2 presents the comparisons between our SDVPRS scheme and Wei et al.'s DVPRS scheme [14] in terms of the computational overheads of the **Sign**, **ReSign**, **ReVer** and **Verify** algorithms. It should be noted that $g^{x_S y_S}$ and $e(g^{x_S}, g^{y_S})^{x_V}$ can be pre-computed in our scheme. To generate a signature, our scheme needs two exponentiations and one bilinear pairing operation (29.05 ms), while Wei et al.'s scheme [14] needs one exponentiation and one bilinear pairing operation (21.59 ms). As for the computation cost of signature conversion, our scheme requires two exponentiations (14.92 ms), while Wei et al.'s scheme [14] requires one exponentiation (7.46 ms). For the signature verification process, our scheme needs one exponentiation and one bilinear pairing operation (21.59 ms), while Wei et al.'s scheme [14] requires one exponentiation and two bilinear pairing operations (35.72 ms).

From the above analysis, we have observed that our scheme is comparable in computational performance to that of Wei et al.'s scheme [14]. However, our SDVPRS scheme provides higher security because the security of our scheme does not depend on ideal random oracles.

6. Application for Environmental Monitoring System

We describe an environmental monitoring data transmission system using our proposed SDVPRS scheme. This system, as shown in Figure 2, consists of three entities: IoT device, a server node and a set of n data centers that store IoT data.

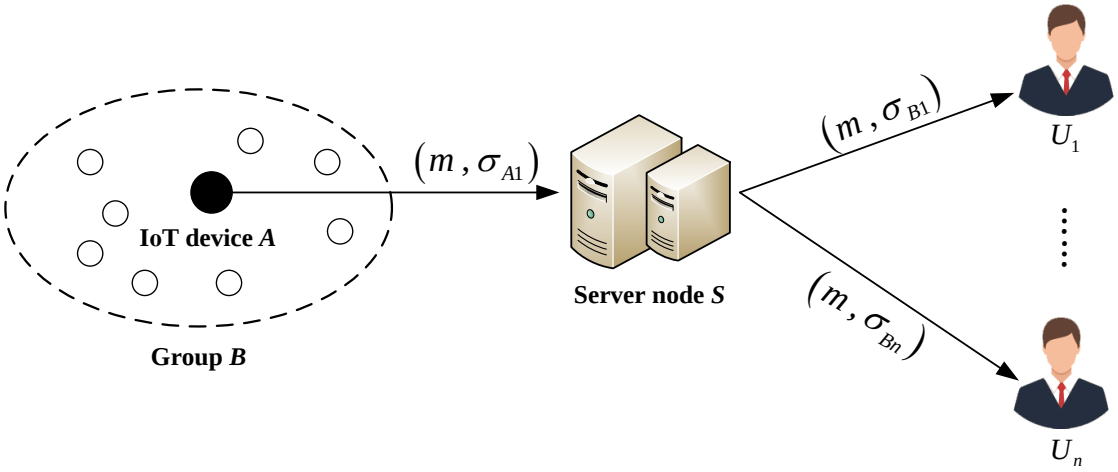

Figure 2. Environmental monitoring data transmission based on IoT.

The IoT device A belonging to a group B is mainly responsible for collecting environmental data and generating signatures of these data. Each data center U_i has powerful computing and storage capabilities to store or analyze environmental data sent by IoT devices. For example, data centers receiving environmental data are usually government agencies, universities, research institutes or various types of environmental companies. The server node S acts as a proxy with a certain computing power and communication capabilities. It can convert the signature of IoT device A into the signature of group B, where the IoT device is located, and send the environmental monitoring data and corresponding signature anonymously to a subset of some designated data centers (denoted by $\{U_1, ..., U_n\}$). The system consists of four phases: system setup, data acquisition, data transmission and data storage. The environmental monitoring data transmission system is described as follows.

- System setup:

  The system parameters $sp = (G_1, G_2, p, g, e, u_0, u_1, ..., u_n)$ are generated by the **Setup** algorithm in Section 4. Each entity in the system generates its public/secret key pair (pk, sk) by running the **KeyGen** algorithm in Section 4.

  1. The IoT device A sets its secret key $sk_A = (sk_{A,1}, sk_{A,2}) = (x_A, y_A)$ and public key $pk_A = (pk_{A,1}, pk_{A,2}) = (g^{x_A}, g^{y_A})$. Similarly, the group B in which the IoT device A is located sets the secret key $sk_B = (sk_{B,1}, sk_{B,2}) = (x_B, y_B)$ and public key $pk_B = (pk_{B,1}, pk_{B,2}) = (g^{x_B}, g^{y_B})$.
  2. Each data center U_i sets its public/secret key pair $(pk_i, sk_i) = (g^{x_i}, x_i), i = 1, ..., n$.
  3. The server node S runs the **ReSKey** algorithm in Section 4 to generate the re-signing key $rsk_{A \to B} = \frac{x_B y_B}{x_A y_A} (\text{mod } p)$ between the IoT device A and the group B. The server node S then runs the **ReVKey** algorithm in Section 4 to generate the re-designate-verifier key $rvk_{1 \to i} = \frac{x_i}{x_1} (\text{mod } p)$ between the data center U_1 and the other data center $U_i (i = 2, ..., n)$.

- Data acquisition:

  The IoT device A that assembles the sensor(s) collects environmental monitoring data $m = (m_1, ...m_n) \in \{0,1\}^n$ (e.g., atmosphere, water, soil, plants, etc.). Then, A randomly selects $r \in Z_p$ and calculates $\sigma_{A1,2} = g^r$ and $\sigma_{A1,1} = e(g^{x_A y_A}(u_0 \prod_{i=1}^{n} u_i^{m_i})^r, g^{x_1})$, where g^{x_1} is the public key of the data center U_1. Finally, A sends m and the corresponding signature $\sigma_{A1} = (\sigma_{A1,1}, \sigma_{A1,2})$ to the server node S.

- Data transmission:

  After receiving the message m and signature $\sigma_{A1} = (\sigma_{A1,1}, \sigma_{A1,2})$ sent by A, the server node S performs the following steps.

  1. S calculates the signature $\sigma_{B1} = (\sigma_{B1,1}, \sigma_{B1,2}) = ((\sigma_{A1,1})^{rsk_{A\to B}}, (\sigma_{A1,2})^{rsk_{A\to B}})$ using the re-signature key $rsk_{A\to B}$. That is, the server node S converts the signature σ_{A1} of the IoT device into the signature σ_{B1} on m of the group B.
  2. S uses the re-designate-verifier key $rvk_{1\to i}$ to convert the signature $\sigma_{B1} = (\sigma_{B1,1}, \sigma_{B1,2})$ to $\sigma_{Bi} = (\sigma_{Bi,1}, \sigma_{Bi,2})$, where $\sigma_{Bi,1} = (\sigma_{B1,1})^{rvk_{1\to i}}$ and $\sigma_{Bi,2} = (\sigma_{B1,2})^{rvk_{1\to i}}$ for $i = 2, ..., n$.
  3. S sends m and the corresponding signature σ_{Bi} to the designated data center U_i, $i = 1, ..., n$.

- Data storage:

  After receiving the environmental monitoring data $m = (m_1, ...m_n) \in \{0,1\}^n$ and the corresponding signature $\sigma_{Bi} = (\sigma_{Bi,1}, \sigma_{Bi,2})$, each data center $U_i (1 \le i \le n)$ uses its own private key $sk_i = x_i$ and the public key $pk_B = (g^{x_B}, g^{y_B})$ of group B to check whether the following equation holds:

$$\sigma_{Bi,1} = e(g^{x_B}, g^{y_B})^{x_i} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_{Bi,2})^{x_i}.$$

  If the above equation does not hold, it means that m is not authentic environmental monitoring data, and U_i aborts and stops. Otherwise, m is securely stored by the data center U_i so that other authorized users can access and analyze the collected environmental data.

In the above system, only the data center U_i can verify the validity of the signature σ_{Bi} on the environmental monitoring data m. The server node S converts the signature of the IoT device A into the signature of the group B, so that each data center U_i cannot infer the true identity of the IoT device from the received signature. Therefore, our system can verify the authenticity and integrity of environmental monitoring data, while protecting the identity privacy of IoT devices.

Here, we experimentally evaluate the performance of the proposed system. The experimental environment is the same as the simulation environment in Section 5. In the data acquisition phase, the following values can be precomputed:

$$Z_1 = g^{x_A y_A}, \quad Z_2 = g^r, \quad T_0 = u_0^r, \quad T_i = u_i^r, \quad i = 1, ..., n.$$

For environmental monitoring data $m = (m_1, ...m_n) \in \{0,1\}^n$, IoT device A generates a signature $\sigma_{A1} = (\sigma_{A1,1}, \sigma_{A1,2})$ on m as:

$$\sigma_{A1} = (e(Z_1 \cdot T_0 \cdot \prod_{i=1}^{n} T_i^{m_i}, pk_1), Z_2),$$

where $pk_1 = g^{x_1}$ is the public key of data center U_1. Figure 3 shows that when the length of the data is 100 bits, 300 bits, 500 bits, 700 bits and 900 bits, the time overhead of the IoT device is up to 20 ms, 27 ms, 35 ms, 42 ms and 50 ms, respectively.

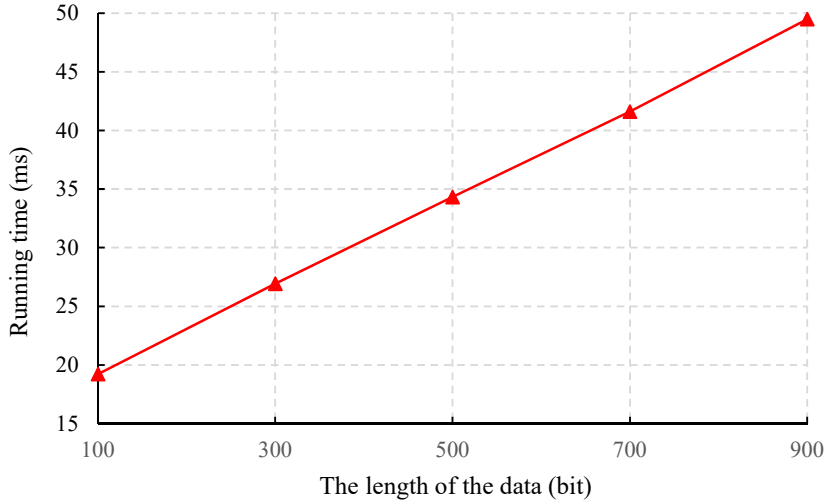

Figure 3. Time cost of the IoT device.

In the data transmission phase, the server node needs two exponentiations to convert the signature σ_{A1} of the IoT device into the signature σ_{B1} of its group. In addition, the server node needs two exponentiations to change the data center U_1 in σ_{B1} to the data center U_i. Figure 4 shows that when the total number of data centers is 2, 4, 6, 8 and 10, the computational overhead required by the server node is approximately 29 ms, 60 ms, 92 ms, 125 ms and 156 ms, respectively.

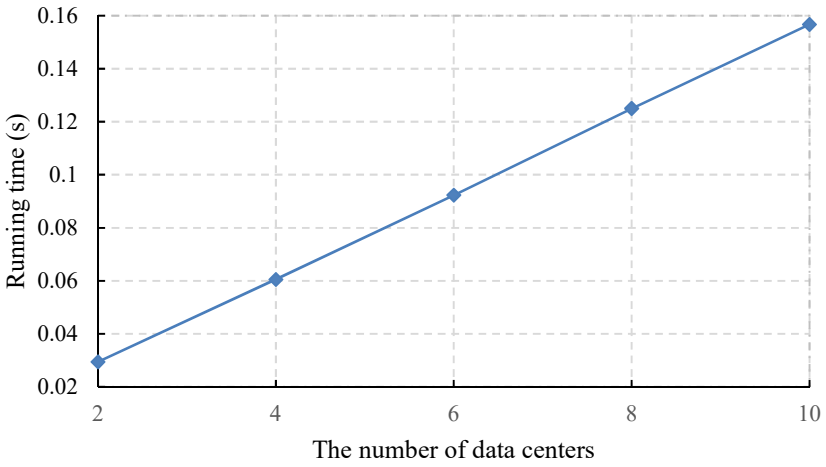

Figure 4. Time cost of the server node.

For the data storage phase, the signature verification equation of the data is:

$$\sigma_{Bi,1} = e(g^{x_B}, g^{y_B})^{x_i} e(u_0 \prod_{i=1}^{n} u_i^{m_i}, \sigma_{Bi,2})^{x_i},$$

where $e(g^{x_B}, g^{x_B})^{x_i}$ can be pre-computed. When the length of the data is 100 bits, 300 bits, 500 bits, 700 bits and 900 bits, the time overhead required for each data center is as shown in Figure 5. Because the server nodes and data centers have strong computing power, our proposal has practical application.

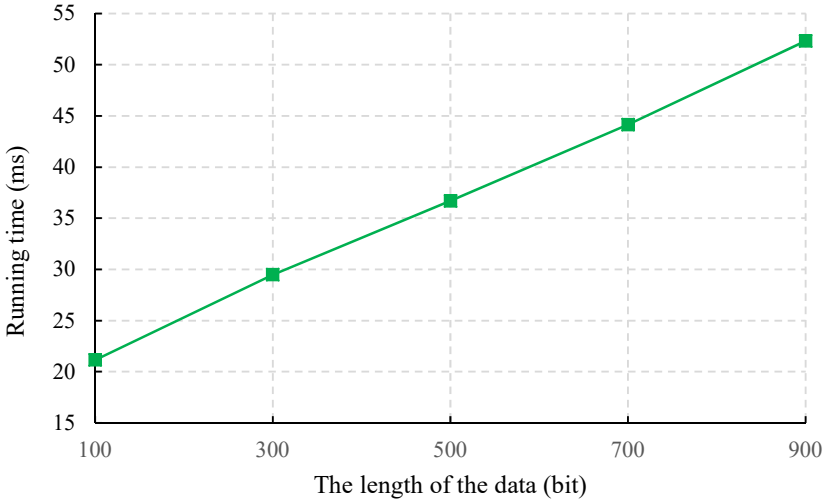

Figure 5. Time cost for verifying signature validity in each data center.

7. Conclusions

Data authenticity and identity privacy are still critical issues for IoT devices. To secure IoT devices, a new SDVPRS technique applied to the IoT environment is presented in this paper. First, we present the security concepts of SDVPRS, and then, we propose the first construction of an SDVPRS scheme without random oracles. Furthermore, we prove that the proposed scheme is secure in the standard model based on the BDH, GBDH and DBDH problems. The security proofs demonstrate that our SDVPRS scheme can protect the identity privacy of IoT devices while ensuring the authenticity and integrity of IoT data. Our SDVPRS scheme is very useful for IoT-based data transmission systems.

Author Contributions: For the research paper, X.-D.Y. and L.-K.X. proposed and designed the SDVPRS scheme. C.-L.C. presented the background about SDVPRS. C.-F.W. did the performance analysis. The four authors cooperatively proved the security of the proposed scheme.

Funding: This research was funded by the National Natural Science Foundation of China Grant Numbers 61662069, 61672020 and 61472433; the China Postdoctoral Science Foundation Grant Number 2017M610817; the Science and Technology Project of Lanzhou City of China Grant Number 2013-4-22; and the Foundation for Excellent Young Teachers by Northwest Normal University Grant Number NWNU-LKQN-14-7.

Acknowledgments: The authors would like to appreciate the anonymous referees for their valuable comments and constructive suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, Y.; Wu, L.; Yin, G.; Li, L.; Zhao, H. A survey on security and privacy issues in internet-of-things. *IEEE Internet Things J.* **2017**, *4*, 1250–1258. [CrossRef]
2. Shen, L.; Ma, J.; Liu, X.; Wei, F.; Miao, M. A secure and efficient id-based aggregate signature scheme for wireless sensor networks. *IEEE Internet Things J.* **2017**, *4*, 546–554. [CrossRef]
3. Jia, X.; He, D.; Liu, Q.; Choo, K.K.R. An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment. *Ad Hoc Netw.* **2018**, *71*, 78–87. [CrossRef]
4. Kumar, P.; Kumari, S.; Sharma, V.; Sangaiah, A.K.; Wei, J.; Li, X. A certificateless aggregate signature scheme for healthcare wireless sensor network. *Sustain. Comput. Inform. Syst.* **2018**, *18*, 80–89. [CrossRef]
5. Wang, L.; Chen, K.; Long, Y.; Wang, H. An efficient pairing-free certificateless signature scheme for resource-limited systems. *Sci. China Inf. Sci.* **2017**, *60*, 119102. [CrossRef]
6. Yeh, K.H.; Su, C.; Choo, K.K.R.; Chiu, W. A novel certificateless signature scheme for smart objects in the Internet-of-Things. *Sensors* **2017**, *17*, 1001. [CrossRef] [PubMed]

7. Blaze, M.; Bleumer, G.; Strauss, M. Divertible protocols and atomic proxy cryptography. In Proceedings of the Theory and Applications of Cryptographic Techniques (EUROCRYPT'98), Espoo, Finland, 31 May–4 June 1998; Springer: Berlin/Heidelberg, Germany, 1996; LNCS Volume 1403, pp. 127–144.
8. Yang, X.; Chen, C.; Ma, T.; Wang, J.; Wang, C. Revocable identity-based proxy re-signature against signing key exposure. *PLoS ONE* **2018**, *13*, e0194783. [CrossRef] [PubMed]
9. Shao, J.; Cao, Z.; Wang, L.; Liang, X. Proxy re-signature schemes without random oracles. In Proceedings of the Cryptology in India (INDOCRYPT), Chennai, India, 9–13 December 2007; Springer: Berlin/Heidelberg, Germany, 2007; LNCS Volume 4859, pp. 197–209.
10. Jakobsson, M.; Sako, K.; Impagliazzo, R. Designated verifier proofs and their applications. In Proceedings of the Theory and Applications of Cryptographic Techniques (EUROCRYPT'96), Saragossa, Spain, 12–16 May 1996; Springer: Berlin/Heidelberg, Germany, 1996; LNCS Volume 1070, pp. 143–154.
11. Zhao, W.; Peng, Y.; Xie, F.; Dai, Z.; Gao, H.; Gao, Y. Designated verifier signature scheme over circulant matrices. In Proceedings of the Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Piraeus, Greece, 18–20 July 2012; IEEE Press: Piscataway, NJ, USA, 2012; pp. 420–423.
12. Li, M.; Fang, T. Provably secure and efficient id-based strong designated verifier signature scheme with message recovery. In Proceedings of the Network-Based Information Systems (NBiS), Salerno, Italy, 10–12 September 2014; IEEE Press: Piscataway, NJ, USA, 2014; pp. 287–293.
13. Tso, R.; Nieto, J.M.G.; Okamoto, T.; Boyd, C.; Okamoto, E. Verifier-key-flexible universal designated-verifier signatures. In Proceedings of the Cryptography and Coding, Cirencester, UK, 18–20 December 2007; Springer: Berlin/Heidelberg, Germany, 2007; LNCS Volume 4887, pp. 403–421.
14. Wei, J.; Yang, G.; Mu, Y. Designated verifier proxy re-signature for deniable and anonymous wireless communications. *Wirel. Pers. Commun.* **2017**, *97*, 3017–3030. [CrossRef]
15. Yang, X.; Yang, M.; An, F.; Leng, Q. A payment mechanism with multi-authority and privacy protection in mobile social networks. In Proceedings of the Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, 15–17 December 2017; IEEE Press: Piscataway, NJ, USA, 2017; pp. 258–262.
16. Zhou, C.X. Identity based generalized proxy signcryption scheme. *Inf. Technol. Control* **2016**, *45*, 13–26.
17. Hu, X.; Tan, W.; Xu, H.; Wang, J. Short and provably secure designated verifier proxy signature scheme. *IET Inf. Secur.* **2016**, *10*, 69–79. [CrossRef]
18. Canetti, R.; Goldreich, O.; Halevi, S. The random oracle methodology, revisited. *J. ACM (JACM)* **2004**, *51*, 557–594. [CrossRef]
19. Waters, B. Efficient identity-based encryption without random oracles. In Proceedings of the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2005), Aarhus, Denmark, 22–26 May 2005; LNCS Volume 3494, pp. 114–127.
20. Gao, C.Z.; Cheng, Q.; He, P.; Susilo, W.; Li, J. Privacy-preserving Naive Bayes classifiers secure against the substitution-then-comparison attack. *Inf. Sci.* **2018**, *444*, 72–88. [CrossRef]
21. Li, T.; Li, J.; Liu, Z.; Li, P.; Jia, C. Differentially private naive bayes learning over multiple data sources. *Inf. Sci.* **2018**, *444*, 89–104. [CrossRef]
22. Zhang, X.; Tan, Y.; Liang, C.; Li, Y.; Li, J. A covert channel over volte via adjusting silence periods. *IEEE Access* **2018**, *6*, 9292–9302. [CrossRef]
23. Huang, Z.; Liu, S.; Mao, X.; Chen, K.; Li, J. Insight of the protection for data security under selective opening attacks. *Inf. Sci.* **2017**, *412*, 223–241. [CrossRef]
24. Li, J.; Chen, X.; Chow, S.S.; Huang, Q.; Wong, D.S.; Liu, Z. Multi-authority fine-grained access control with accountability and its application in cloud. *J. Netw. Comput. Appl.* **2018**, *112*, 89–96. [CrossRef]
25. Castiglione, A.; De Santis, A.; Masucci, B.; Palmieri, F.; Castiglione, A.; Li, J.; Huang, X. Hierarchical and shared access control. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 850–865. [CrossRef]
26. Shen, H.; Gao, C.; He, D.; Wu, L. New biometrics-based authentication scheme for multi-server environment in critical systems. *J. Ambient Intell. Humaniz. Comput.* **2015**, *6*, 825–834. [CrossRef]
27. Cai, Z.; Yan, H.; Li, P.; Huang, Z.A.; Gao, C. Towards secure and flexible EHR sharing in mobile health cloud under static assumptions. *Clust. Comput.* **2017**, *20*, 2415–2422. [CrossRef]
28. Shen, J.; Zhou, T.; Chen, X.; Li, J.; Susilo, W. Anonymous and traceable group data sharing in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 912–925.[CrossRef]

29. Lin, Q.; Yan, H.; Huang, Z.; Chen, W.; Shen, J.; Tang, Y. An ID-based linearly homomorphic signature scheme and its application in blockchain. *IEEE Access* **2018**, *6*, 20632–20640. [CrossRef]
30. Chen, W.; Lei, H.; Qi, K. Lattice-based linearly homomorphic signatures in the standard model. *Theor. Comput. Sci.* **2016**, *634*, 47–54. [CrossRef]
31. Lin, Q.; Li, J.; Huang, Z.; Chen, W.; Shen, J. A short linearly homomorphic proxy signature scheme. *IEEE Access* **2018**, *6*, 12966–12972. [CrossRef]
32. Tian, H.; Li, J. A short non-delegatable strong designated verifier signature. *Front. Comput. Sci.* **2014**, *8*, 490–502. [CrossRef]
33. Chen, W.; Chen, Z.; Samatova, N.F.; Peng, L.; Wang, J.; Tang, M. Solving the maximum duo-preservation string mapping problem with linear programming. *Theor. Comput. Sci.* **2014**, *530*, 1–11. [CrossRef]
34. Ateniese, G.; Hohenberger, S. Proxy re-signatures: New definitions, algorithms, and applications. In Proceedings of the Computer and Communications Security (CCS), Alexandria, VA, USA, 7–11 November 2005; ACM Press: New York, NY, USA, 2005; pp. 310–319.
35. Hu, X.; Zhang, Z.; Yang, Y. Identity based proxy re-signature schemes without random oracle. In Proceedings of the Computational Intelligence and Security (CIS), Beijing, China, 11–14 December 2009; IEEE Press: Piscataway, NJ, USA, 2009; pp. 256–259.
36. Tian, M. Identity-based proxy re-signatures from lattices. *Inf. Process. Lett.* **2015**, *115*, 462–467. [CrossRef]
37. Wang, Z.W.; Xia, A.D. ID-based proxy re-signature with aggregate property. *J. Inf. Sci. Eng.* **2015**, *31*, 1199–1211.
38. Yang, P.; Cao, Z.; Dong, X. Threshold proxy re-signature. *J. Syst. Sci. Complex.* **2011**, *24*, 816–824. [CrossRef]
39. Yang, X.D.; Wang, C.F.; Lan, C.H.; Wang, B. Flexible threshold proxy re-signature schemes. *Chin. J. Electron.* **2011**, *20*, 691–696.
40. Yang, X.; Wang, C.; Zhang, L.; Qiu, J. On-line/off-line threshold proxy re-signatures. *Chin. J. Electron.* **2014**, *23*, 248–253.
41. Guo, D.; Wei, P.; Yu, D.; Yang, X. A certificateless proxy re-signature scheme. In Proceedings of the Computer Science and Information Technology (ICCSIT), Chengdu, China, 9–11 July 2010; IEEE Press: Piscataway, NJ, USA, 2010; pp. 157–161.
42. Xiao, H.; Zhang, M. Provably-secure certificateless proxy re-signature scheme. In Proceedings of the Intelligent Networking and Collaborative Systems (INCoS), Xi'an, China, 9–11 September 2013; IEEE Press: Piscataway, NJ, USA, 2013; pp. 591–594.
43. Hu, X.; Liu, Y.; Xu, H.; Wang, J.; Zhang, X. Analysis and improvement of certificateless signature and proxy re-signature schemes. In Proceedings of the Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 19–20 December 2015; IEEE Press: Piscataway, NJ, USA, 2015; pp. 166–170.
44. Saeednia, S.; Kremer, S.; Markowitch, O. An efficient strong designated verifier signature scheme. In Proceedings of the International Conference on Information Security and Cryptology (ICISC), Seoul, Korea, 27–28 November 2003; Springer: Berlin/Heidelberg, Germany, 2003; LNCS Volume 2971, pp. 40–54.
45. Noh, G.; Jeong, I.R. Strong designated verifier signature scheme from lattices in the standard model. *Secur. Commun. Netw.* **2015**, *18*, 6202–6214. [CrossRef]
46. Chen, Y.; Zhao, Y.; Xiong, H.; Yue, F. A certificateless strong designated verifier signature scheme with non-delegatability. *IJ Netw. Secur.* **2017**, *19*, 573–582.
47. Khan, A.U.; Ratha, B.K. A secure strong designated verifier signature scheme. *IJ Netw. Secur.* **2017**, *19*, 599–604.
48. Huang, Q.; Yang, G.; Wong, D.S.; Susilo, W. Efficient strong designated verifier signature schemes without random oracles or delegatability. *Int. J. Inf. Secur.* **2009**, *10*, 373–385. [CrossRef]
49. Tian, H.; Jiang, Z.; Liu, Y.; Wei, B. A systematic method to design strong designated verifier signature without random oracles. *Clust. Comput.* **2013**, *16*, 817–827. [CrossRef]
50. Lin, H.Y. Secure universal designated verifier signature and its variant for privacy protection. *Inf. Technol. Control* **2013**, *42*, 268–276.
51. Lin, C.; Wu, W.; Huang, X.; Xu, L. A new universal designated verifier transitive signature scheme for big graph data. *J. Comput. Syst. Sci.* **2017**, *83*, 73–83. [CrossRef]
52. Laguillaumie, F.; Vergnaud, D. Multi-designated verifiers signatures: Anonymity without encryption. *Inf. Process. Lett.* **2007**, *102*, 127–132.[CrossRef]

53. Ming, Y.; Wang, Y. Universal designated multi verifier signature scheme without random oracles. *Wuhan Univ. J. Nat. Sci.* **2008**, *13*, 685–691. [CrossRef]
54. Huang, Q.; Yang, G.; Wong, D.S.; Susilo, W. Efficient strong designated verifier signature schemes without random oracle or with non-delegatability. *Int. J. Inf. Secur.* **2011**, *10*, 373–385. [CrossRef]