Efficient and General PVSS Based on ElGamal Encryption

Kun Peng*

Abstract—PVSS stands for publicly verifiable secret sharing. In PVSS, a dealer shares a secret among multiple share holders. He encrypts the shares using the shareholders' encryption algorithms and publicly proves that the encrypted shares are valid. Most of the existing PVSS schemes do not employ an ElGamal encryption to encrypt the shares. Instead, they usually employ other encryption algorithms like a RSA encryption and Paillier encryption. Those encryption algorithms do not support the shareholders' encryption algorithms to employ the same decryption modulus. As a result, PVSS based on those encryption algorithms must employ additional range proofs to guarantee the validity of the shares obtained by the shareholders. Although the shareholders can employ ElGamal encryptions with the same decryption modulus in PVSS such that the range proof can be avoided, there are only two PVSS schemes based on ElGamal encryption. Moreover, the two schemes have their drawbacks. One of them employs a costly repeating-proof mechanism, which needs to repeat the dealer's proof at least scores of times to achieve satisfactory soundness. The other requires that the dealer must know the discrete logarithm of the secret to share and thus weakens the generality and it cannot be employed in many applications. A new PVSS scheme based on an ElGamal encryption is proposed in this paper. It employs the same decryption modulus for all the shareholders' ElGamal encryption algorithms, so it does not need any range proof. Moreover, it is a general PVSS technique without any special limitation. Finally, an encryption-improving technique is proposed to achieve very high efficiency in the new PVSS scheme. It only needs a number of exponentiations in large cyclic groups that are linear in the number of the shareholders, while all the existing PVSS schemes need at least a number of exponentiations in large cyclic groups that are linear in the square of the number of the shareholders

Keywords—ElGamal, PVSS

1. INTRODUCTION

Secret sharing is a technique to share a secret among some shareholders and allow certain subsets of them to reconstruct the secret. It is needed in applications where no single entity is trusted and an important power or capability must be shared among multiple parties. The most popular secret sharing mechanism is the t-out-of-n threshold secret sharing [12], where a dealer shares a secret s among n shareholders and allows any t of them to reconstruct it.

Manuscript received October 11, 2011; accepted January 10, 2012.
Corresponding Author: Kun Peng
* Institute for Inforcomm Research, Singapore (dr.kun.peng@gmail.com)

The dealer builds a polynomial $f(x) = \sum_{j=0}^{t-1} f_j x^j$ where $f_0 = s$ and $f_1, f_2, \ldots, f_{t-1}$ are random integers. He then sends $s_i = f(i)$ to the i^{th} shareholder for $i = 1,2,\ldots,n$. Any t shares can be used to reconstruct the secret using Lagrange Interpolation, while no information about the secret is obtained if the number of available shares is smaller than t. Secret sharing is widely employed in various secure protocols like e-auctions, e-voting, and multiparty computation. As most of the applications require public verifiability, very often secret sharing must be publicly verifiable. Namely, it must be publicly verified without revealing the secret or any of its shares where all of the n shares are consistently generated from a unique secret-generating polynomial such that any t of them reconstructs the same secret.

Publicly verifiable secret sharing is usually called PVSS. Famous PVSS schemes include [3, 8, 10, 13], where one of the two PVSS protocols in [13] is a development of the proposal in [7]. The most recently published PVSS scheme seems to be [9]. PVSS is actually a combination of secret sharing and publicly verifiable encryption and usually works as follows:

1. The dealer encrypts the shares for the share holders using their public keys and publishes the ciphertexts. Suppose $E_i()$ is the i^{th} shareholder's public encryption algorithm, then $c_i = E_i(s_i)$ for $i = 1,2,\ldots,n$ are published. Each share holder can decrypt the ciphertext for him and obtain his share.
2. The dealer commits to the secret-generating polynomial (including all its coefficients) and publishes the commitment.
3. The dealer publicly proves that each encrypted share is generated by the committed secret-generating polynomial.

The following three important security properties are desired in PVSS.

- Correctness: if the dealer is honest and does not deviate from the PVSS protocol, he can always successfully prove the validity of the shares.
- Soundness: if the dealer's proof of validity of the shares is passed with a non-negligible probability, it is guaranteed that they are generated by the committed secret-generating polynomial such that any t of them reconstructs the same secret.
- Privacy: no information about the secret or any of its shares is revealed in the proof of validity of the shares. More precisely, a private PVSS scheme should employ zero knowledge proof techniques, which do not reveal any secret information.

An important phenomenon should be noticed in PVSS: $D_i()$, the decryption function for $E_i()$, has a decryption modulus p_i. More precisely, instead of guaranteeing $D_i(E_i(M)) = M$, it actually guarantees $D_i(E_i(M)) = M \bmod p_i$ where p_i is the modulus used in $D_i()$ to calculate its decryption result. So the i^{th} shareholder actually obtains $s_i \bmod p_i$ instead of s_i by decrypting c_i. If $s_i < p_i$ he obtains the same share; otherwise he obtains a different share. As $f_1, f_2, \ldots, f_{t-1}$ are randomly chosen in the share-generating polynomial, it is possible that $s_i \geq p_i$. Even if it is required to choose $f_1, f_2, \ldots, f_{t-1}$ under a threshold or employ a modulus in the share-generating polynomial to prevent s_i from overflowing p_i, a malicious dealer can ignore the requirements and generate s_i larger than p_i. If $p_1 = p_2 = \ldots = p_n$. A

possible change of the shares when being decrypted is not a problem as the same modulus can be used in share generation and share reconstruction. With all the computations in the PVSS scheme employing the same modulus, changing any integer in any computation by adding or subtracting a multiple of the modulus does not affect the correctness of the computations. When the share holders' encryption algorithms employ different decryption moduli, shares overflowing the moduli is a serious problem in security as even if the dealer encrypts a share of the secret for a shareholder and successfully proves that the encrypted share can be used to reconstruct the secret correctly the shareholder may obtain an incorrect share.

Unfortunately, in the commonly used public key encryption algorithms, only ElGamal encryption supports its different instances to employ the same decryption modulus. The other algorithms like RSA encryption or Paillier encryption must employ different decryption moduli for different users, as their security depends on the hardness of the factorization problem, and so they must employ different composite moduli in encryption and thus use different decryption moduli. So if those encryption algorithms are employed in PVSS, every s_i encrypted in c_i must be proved by the dealer to be smaller than p_i, otherwise the validity of the shares cannot be guaranteed. Namely, n instances of the so-called range proof (defined in [4]) are needed in the PVSS schemes employing those encryption algorithms. Such PVSS schemes include [3, 8, 9], which are inefficient.

The only PVSS schemes employing ElGamal encryption for the shareholders are [13] and [10]. Although they can employ the same decryption modulus for all the shareholders' encryption algorithms and are inherently immune to any problem caused by an overflow of the shares, they have their own drawbacks as explained in Section 2. Actually, all the existing PVSS schemes are either limited in application or inefficient in computation as explained in Section 2.

In this paper, a new PVSS scheme based on ElGamal encryption is proposed. It employs the same decryption modulus for all the shareholders' ElGamal encryption algorithms so that there is no worry about an overflow of the encrypted shares and no range proof is needed. Moreover, it is a general PVSS technique without any special limitation and so it can be employed in any application of PVSS. Finally, an encryption-improving technique is proposed to achieve very high efficiency in the new PVSS scheme. It only needs $O(n)$ exponentiations in large cyclic groups and is much more efficient than the existing PVSS schemes, which need at least $O(tn)$ exponentiations in large cyclic groups in computation where t usually has the same magnitude as n.

2. LIMITED APPLICATION AND INEFFICIENCY OF THE EXISTING PVSS SCHEMES

The existing PVSS schemes [3, 8-10, 13] are either limited in application or inefficient in computation or have both drawbacks. More precisely, the PVSS schemes in [10 and 9] and the two PVSS protocols in [3] with delayed recovery depend on special conditions and thus are not general PVSS schemes that are suitable for various applications, while the PVSS schemes in [3, 8, 13] are inefficient in computation.

In the PVSS scheme in [10], a dealer employs a special sharing function to share a specially generated secret, while the other PVSS schemes employ the normal sharing function by Shamir [12] and can share any secret in general. Moreover, a corresponding special secret reconstruction

function is employed in [10] to reconstruct the secret accordingly. The special sharing function reconstruction function in [10] is described as follows:

- Sharing
  (a) The dealer first chooses δ and then calculates the secret $s = g^{\delta}$ where g is the generator of a large cyclic group.
  (b) He builds a polynomial $f(x) = \sum_{j=0}^{t-1} a_j x^j$ with $a_0 = \delta$ and a_j for $j = 1,2,\ldots,t-1$ are random integers.
  (c) He publishes the ElGamal ciphertext $c_i = (g^{r_i}, g^{\delta_i} y_i^{r_i})$ for $i = 1,2,\ldots,n$ where $\delta_i = f(i)$, y_i is P_i's ElGamal public key.

- Reconstruction
  (a) Each P_i decrypt c_i and obtains $s_i' = g^{\delta_i}$.
  (b) A set with at least t sharers are put together: $s = \prod_{i \in I} s_i'^{u_i}$ where $u_i = \prod_{j \in I, j \neq i} j/(j-i)$ and I is the set containing the indices of the t shares.

In the PVSS scheme in [10], $a_0 = \delta$, so actually the discrete logarithm of the secret s is shared using the share-generating polynomial. Its reconstruction function is accordingly changed to reconstruct s using the shares of δ. So knowledge of a discrete logarithm of the secret is compulsory to the dealer in the PVSS scheme in [10]. Therefore, it is not suitable for some applications. One of the most common applications of PVSS is key sharing (also known as distributed key generation in some cryptographic schemes). To the best of our knowledge, a secret key is usually chosen from a consecutive interval range in normal cryptographic schemes and no cryptographic scheme first chooses a discrete logarithm of the secret key and then calculates the secret key in a cyclic group. Actually, we do not know of any key generation algorithm that raises a generator of a cyclic group to the power of a chosen integer to generate a key in a cyclic group. Another important application of PVSS is the sharing of passwords or accessing codes in a distributed access control. In most applications, users usually randomly choose a password or access a code and very often the users would like to choose some special password or access a code like their birthdays or phone number. So it is very probable that a discrete logarithm of the password or access code is unknown or even does not exist at all.

If the dealer does not know the discrete logarithm of the secret in [10], its PVSS scheme can only work like the two PVSS protocols with delayed recovery in [3]. However, PVSS with delayed recovery cannot distribute a secret to the shareholders in real time, as the shareholders in the PVSS system need to search for discrete logarithms in a large range to obtain their shares. So they are inefficient in secret reconstruction and are not suitable for many applications where share retrieval or secret reconstruction cannot be delayed. The PVSS scheme in [9] requires the dealer to prove that each share s_i is in the range $\{0,1,\ldots,N_i\}$ where N_i is the multiplica-

tive modulus used in the decryption function of P_i's encryption algorithm. Otherwise, reconstruction of the secret may go wrong due to an attack in the end of Section 4 in [9]. However, the range proofs in [9] can only work under a special condition. In Section 5 of [9], to guarantee that s_i is in the range $\{0,1,\ldots,N_i\}$, it is proved that $s_i + s_i' = N_i$ where s_i' is another integer with unverified distribution. Obviously, $s_i + s_i' = N_i$ alone cannot guarantee that s_i is in the range $\{0,1,\ldots,N_i\}$. When s_i is out of the range, it is changed when its encryption is decrypted and any secret reconstruction using it will fail. So security of the PVSS scheme in [9] depends on distribution of the shares and is not always satisfied.

It has been shown in [3, 8] that their general PVSS protocols can choose 3 as the RSA public keys of the shareholders to improve efficiency. However, too small of a RSA public key, like 3 or 5, is usually impractical in real-life cryptographic protocols and it is widely believed in the cryptographic community that smaller public keys make the RSA cipher more vulnerable to attacks, especially when a proper padding of the message is absent. A famous example is that in a broadcasting protocol an attack can be launched if multiple receivers use the same small public key e (e.g., 3) and employ different RSA moduli $N_1, N_2, N_3, \ldots$. When message m is broadcast, it is in the form $m^e \bmod N_1$, $m^e \bmod N_2$, $m^e \bmod N_3 \ldots$. When multiple ciphertexts like these are put together and the product of their multiplicative moduli in encryption is larger than m^e in Z, the Chinese Remainder Theorem can be employed to obtain m^e in Z and then m can be extracted. The smaller e is, the fewer ciphertexts are needed and the attack is easier. For example, when $e = 3$, three ciphertexts $m^e \bmod N_1$, $m^e \bmod N_2$ and $m^e \bmod N_3$ are enough for the attack.

A more formal and detailed analysis of the vulnerability of very small RSA public keys is given in [6], which shows that an RSA cipher with too small of a public key like 3 fails, "if the opponent knows two-thirds of the message, or if two messages agree over eight-ninths of their length; and we can find the factors of $N = PQ$ if we are given the high order bits of P." Even those cryptographic schemes aiming at improving the efficiency of an RSA cipher like [2] agree that, "RSA as usually deployed uses a larger public exponent (e = 65537)." So, in authoritative security standards like the NIST standard [1], it is required that the public key of an RSA cipher must be no smaller than 65537.

In PVSS, as publicly verifiable encryption is necessary, secure padding of a message is impossible (or at least very difficult). So the threat of an attack exploiting small RSA public keys is serious in PVSS and the public keys of the RSA cipher in PVSS should be more cautiously chosen. Therefore, in practical PVSS applications, not only should too small public keys, like 3 or 5, be avoided but also a more strict security standard should be followed and larger RSA public keys should be adopted. When the public keys of the shareholders $e_1, e_2, \ldots, e_n$ are not too small (e.g., when they are no smaller than 65537), the PVSS scheme in [8] and the two general PVSS protocols in [3] (with fast recovery) are inefficient, as they cost $O(\sum_{i=1}^{n}(t + 8d_H(\log_2 e_i))$ expo-

nentiations in large cyclic groups where $d_H(\log_2 e_i)$ is the Hamming distance of e_i.

The PVSS protocols in [13] repeat their proof of validity for every encrypted share scores of times to obtain practical soundness. When they are repeated K times, the validity of the shares can be guaranteed except for a probability 2^{-K}. To achieve satisfactory soundness, their computational cost $O(Ktn)$ is very high.

3. NEW PVSS BASED ON AN ELGAMAL ENCRYPTION

In our proposal the shareholders are denoted as $P_1, P_2, \ldots, P_n$. A unique decryption modulus N is used in PVSS where $N = pq$ and $p = 2p' + 1$ and $q = 2q' + 1$ are strong primes and $GCD(N, \varphi(N)) = 1$. G is the cyclic subgroup in Z_N^* containing all the quadratic residues and g is its generator. Each shareholder P_i sets up his ElGamal encryption algorithm. P_i randomly chooses his private key x_i in Z_N^* and publishes his public key $y_i = g^{x_i} \bmod N$. The secret sharing and reconstruction operations are as follows, where the secret to share is an integer s in Z_N:

• Share generation and distribution

(a) The dealer randomly chooses integers $f_1, f_2, \ldots, f_{t-1}$ from Z_N and generates a polynomial $F(x) = \sum_{j=0}^{t-1} f_j x^j$ where $f_0 = s$.

(b) The dealer generates $s_i = F(i) \bmod N$ as P_i's share for $i = 1, 2, \ldots, n$.

(c) The dealer sends $c_i = E_i(s_i) = (a_i, b_i) = (g^{r_i} \bmod N, s_i y_i^{r_i} \bmod N)$ to P_i for $i = 1, 2, \ldots, n$ where r_i is randomly chosen from Z_N.

• Secret reconstruction

(a) Each shareholder A_i decrypts c_i and obtains its share s_i.

(b) Any t sharers can be used to reconstruct the secret: $s = \sum_{i \in I} s_i u_i \bmod N$ where $u_i = \prod_{j \in I, j \neq i} \frac{j}{j - i} \bmod N$ and I contains the indices of the t shares.

When public share verification is required, to prove the validity of the encrypted shares, the dealer has to publish $F_j = g^{f_j} \bmod N$ for $j = 0, 1, \ldots, t-1$. Then he can prove the validity of each c_i as in Fig 1. This new PVSS scheme is a general solution and has no limit in secret generation, secret sharing, or secret reconstruction. So it is suitable for any application of PVSS. Especially, it does not require a discrete logarithm if the secret is chosen before it is generated,

1. The dealer publishes $s_i' = s_i^2 \bmod N$.

2. He proves that s_i' is encrypted in $(a_i^2 \bmod N, b_i^2 \bmod N)$ through the zero knowledge proof of equality of discrete logarithms:

$$\log_g a_i^2 = \log_{y_i} b_i^2 / s_i'$$

by using the zero knowledge proof of equality of discrete logarithms in [5], whi ch also guarantees the prover's knowledge of the discrete logarithm.

3. He proves $D(c_i) = \log_g C_i$ where $C_i = \prod_{j=0}^{t-1} F_j^{i^j} \bmod N$ as follows:

(a) He publishes $C_i' = C_i^{s_i} \bmod N$ and proves:

$$\log_g C_i = \log_{C_i} C_i'$$

using the zero knowledge proof technique in [5], which also guarantees the prov er's knowledge of the discrete logarithm.

(b) He proves knowledge of a secret integer z such that:

$$(g^N)^z g^{s_i'} = C_i' \bmod N$$

where $z = (s_i^2 - s_i')/N$ using zero knowledge proof of the knowledge of the di screte logarithm [11].

4. Anyone can publicly verify the dealer's proof.

Fig. 1. Public proof of the validity of c_i

like in [10], and thus can be employed in applications like distributed key generation and distributed access control. Moreover, it employs ElGamal encryption and thus a uniform decryption modulus for all the shareholders, so it avoids a complex range proof like that in [9]. The new PVSS protocol is more efficient than the PVSS schemes in [3, 8, 9, 13]. It does not employ any repetition mechanism (which is needed in [13]) or inefficient share decryption (which is needed in the PVSS protocols with delayed secret recovery in [3]). Its efficiency does not depend on the key size of the employed encryption algorithm (unlike the PVSS protocols in [3, 8]) or an unguaranteed range proof (unlike the PVSS protocol in [9]).

4. Security Analysis and a Slightly Modified Secret Reconstruction

The correctness of the new PVSS protocol is proved in Theorem 1. Its soundness is proved in Theorem 2, which shows that the validity of the shares is guaranteed in a loose standard, allowing $-s_i$ to be received by P_i. It is enough to guarantee the correct reconstruction of the secret as the slightly modified secret reconstruction function in Fig. 2 demonstrates. As for privacy, the proof protocol in the new PVSS protocol only publishes s_i', C_i, and C_i' besides the underlying zero knowledge proof primitives (including the zero knowledge proof of the knowledge of the discrete logarithm and the zero knowledge proof of the equality of discrete logarithms), which have been formally proved to be zero knowledge and reveal no secret information in [11]

When $D(c_i)$ may be $-\log_g C_i$, the secret reconstruction function can be modified as follows:

1. Each P_i decrypts c_i and obtains the decryption result S_i.

2. Each P_i tests whether:

$$g^{S_i} = C_i \bmod N.$$

If it is satisfied, he sets $s_i = S_i$; otherwise he sets $s_i = -S_i \bmod N$.

3. Any t sharers can be used to reconstruct the secret like in any other secret sharing scheme after the shares are adjusted.

Fig. 2. Modified secret reconstruction

and [5]. As factorization of N is hard, it is hard to retrieve s_i from s_i'. As a discrete logarithm problem is hard, it is difficult to retrieve s_i from C_i or C_i'. So privacy of the new PVSS protocol is guaranteed by the widely believed difficulty of the two computational problems.

Theorem 1. An honest dealer can always strictly follow the new PVSS protocol and pass the verification in the proof protocol in Figure 1.

Proof: If the dealer is honest and strictly follows the new PVSS protocol, the following equations are satisfied:

$$s_i = F(i) = \sum_{j=0}^{t-1} f_j i^j \bmod N \tag{1}$$

$$c_i = E_i(s_i) = (a_i, b_i) = (g^{r_i} \bmod N, s_i y_i^{r_i} \bmod N) \tag{2}$$

$$s_i' = s_i^2 \bmod N \tag{3}$$

$$C_i = \prod_{j=0}^{t-1} F_j^{i^j} \bmod N \tag{4}$$

$$F_j = g^{f_j} \bmod N \tag{5}$$

$$C_i' = C_i^{s_i} \bmod N \tag{6}$$

$$z = (s_i^2 - s_i')/N \tag{7}$$

(2) and (3) imply:

$$\begin{aligned}\log_g a_i^2 &= \log_g (g^{r_i})^2 = \log_g g^{2r_i} = 2r_i \\ &= \log_{y_i} y_i^{2r_i} = \log_{y_i} (s_i y_i^{r_i})^2 / s_i^2 = \log_{y_i} b_i^2 / s_i^2 = \log_{y_i} b_i^2 / s_i'.\end{aligned} \tag{8}$$

(1), (4), (5) and (6) imply:

$$\begin{aligned}\log_g C_i &= \log_g (\prod_{j=0}^{t-1} F_j^{i^j}) = \log_g (\prod_{j=0}^{t-1} (g^{f_j})^{i^j}) \\ &= \sum_{j=0}^{t-1} f_j i^j = s_i = \log_{C_i} (C_i^{s_i}) = \log_{C_i} C_i'\end{aligned} \tag{9}$$

(1), (4), (5), (6) and (7) imply:

$$\begin{aligned}(g^N)^z g^{s_i'} &= (g^N)^{(s_i^2 - s_i')/N} g^{s_i'} = g^{s_i^2 - s_i' + s_i'} = g^{s_i^2} = (g^{s_i})^{s_i} \\ &= (g^{\sum_{j=0}^{t-1} f_j i^j})^{s_i} = (g^{\sum_{j=0}^{t-1} f_j i^j})^{s_i} = (\prod_{j=0}^{t-1} F_j^{i^j})^{s_i} = C_i^{s_i} = C_i' \bmod N\end{aligned} \tag{10}$$

As (8), (9), and (10) are satisfied, the dealer can pass the verification in the proof protocol in Fig 1.

□

Theorem 2. If the dealer passes the verification in the proof protocol in Fig. 1, it is guaranteed that $D(c_i) = \pm \log_g C_i$. Proof: As it is proved that:

$$\log_g a_i^2 = \log_{y_i} b_i^2 / s_i',$$

we can suppose $\log_g a_i^2 = \log_{y_i} b_i^2 / s_i' = r_i'$ and deduce that:

$$\begin{aligned}a_i^2 &= g^{r_i'} \bmod N \\ b_i^2 &= s_i' y_i^{r_i'} \bmod N.\end{aligned}$$

So the message encrypted in $(a_i^2 \bmod N, b_i^2 \bmod N)$ is:

$$b_i^2 / (a_i^2)^{x_i} = s_i' y_i^{r_i'} / g^{r_i' x_i} = s_i' y_i^{r_i'} / y_i^{r_i'} = s_i' \bmod N.$$

Moreover, as:

$$\log_g C_i = \log_{C_i} C_i',$$

we can suppose $\log_g C_i = \log_{C_i} C_i' = \hat{s}_i$ and deduce that:

$$C_i = g^{\hat{s}_i} \bmod N$$
$$C_i' = C_i^{\hat{s}_i} = (g^{\hat{s}_i})^{\hat{s}_i} = g^{\hat{s}_i^2} \bmod N.$$

Finally, as:

$$(g^N)^z g^{s_i'} = C_i' \bmod N,$$

we can deduce that:

$$(g^N)^z g^{s_i'} = g^{\hat{s}_i^2} \bmod N$$

and so:

$$zN + s_i' = \hat{s}_i^2 \bmod order(g).$$

As the dealer has proved his knowledge of z, $\hat{s}_i^2$ and s_i':

$$zN + s_i' = \hat{s}_i^2,$$

otherwise the dealer can calculate in polynomial time the order of g or its multiple, which is impossible as factorization of N is difficult. Therefore:

$$D(c_i) = D(a_i, b_i) = \pm D(a_i^2, b_i^2)^{1/2} = \pm \hat{s}_i = \pm \log_g C_i \bmod N$$

□

5. EFFICIENCY IMPROVEMENT

Although the new PVSS protocol has advantages in efficiency over the existing PVSS schemes, it still needs $O(tn)$ exponentiations in large cyclic groups in computation. The efficiency bottleneck of the new PVSS protocol is a calculation of:

$$C_i = \prod_{j=0}^{t-1} F_j^{i^j} \bmod N \ \ for\ i = 1,2,\ldots,n, \tag{11}$$

which costs tn exponentiations in G, while the other operations only costs $O(n)$ exponentiations in large cyclic groups and exponentiations like i^j are much more efficient. Although the dealer can use his knowledge of s_i to calculate C_i more efficiently as $C_i = g^{s_i} \bmod N$, the shareholders and other observers need to employ (11) to calculate $C_1, C_2, \ldots, C_n$ in their verification of the dealer's proof as they do not know s_i in the verification phase and do not trust the dealer.

To overcome this bottleneck, we exploit a special phenomenon in the new PVSS protocol: in the n instances of calculation of C_i the same t bases $F_0, F_1, \ldots, F_{t-1}$ are used. Although directly calculating $C_1, C_2, \ldots, C_n$ is costly for a verifier, verifying the validity of them can be efficient for him if someone else knows (e.g., by using some other more efficient calculation) and publishes them. In the new PVSS protocol the dealer can calculate and publish $C_i = g^{s_i} \bmod N$ for $i = 1,2,\ldots,n$. Then any other party only needs to verify validity of them efficiently as follows:

1. The verifier randomly chooses integers $\theta_1, \theta_2, \ldots, \theta_n$ from Z_L where L is a large security parameter smaller than p and q.
2. He checks:

$$\prod_{i=1}^{n} C_i^{\theta_i} = \prod_{j=0}^{t-1} F_j^{\sum_{i=1}^{n} \theta_i i^j} \bmod N. \tag{12}$$

3. He accepts the validity of $C_1, C_2, \ldots, C_n$ if (12) holds.

This method only costs $n + t$ exponentiations in large cyclic groups, while as illustrated in Theorem 3, C_i is guaranteed to be $\prod_{j=0}^{t-1} F_j^{i^j} \bmod N$ for $i = 1,2,\ldots,n$ if (12) is satisfied with a non-negligible probability. After this optimisation, the computational cost of the new PVSS protocol is reduced $O(n)$ in terms of exponentiations in large cyclic groups.

Theorem 3. If (12) is satisfied with a probability larger than $1/L$, then it is guaranteed that $C_i = \prod_{j=0}^{t-1} F_j^{i^j} \bmod N$ for $i = 1,2,\ldots,n$. Proof: For any integer l in $\{1,2,\ldots,n\}$ there must exist integers $\theta_1, \theta_2, \ldots, \theta_{l-1}, \theta_{l+1}, \ldots, \theta_n$ in Z_L and two different integers θ_l and $\hat{\theta}_l$ in

Z_L such that:

$$\prod_{i=1}^{n} C_i^{\theta_i} = \prod_{j=0}^{t-1} F_j^{\sum_{i=1}^{n} \theta_i i^j} \bmod N, \tag{13}$$

$$(\prod_{i=1}^{l-1} C_i^{\theta_i}) C_l^{\hat{\theta}_l} \prod_{i=l+1}^{n} C_i^{\theta_i} = \prod_{j=0}^{t-1} F_j^{(\sum_{i=1}^{l-1} \theta_i i^j) + \hat{\theta}_l l^j + \sum_{i=l+1}^{n} \theta_i i^j} \bmod N. \tag{14}$$

Otherwise, with this l for any combination of $\theta_1, \theta_2, \ldots, \theta_{l-1}, \theta_{l+1}, \ldots, \theta_n$ there is at most one θ_l to satisfy (12) among the L possible choices of θ_l, which leads to a contradiction: the probability that (12) is satisfied is no larger than $1/L$. (13)/(14) yields

$$C_l^{\theta_l - \hat{\theta}_l} = \prod_{j=0}^{t-1} F_j^{(\theta_l - \hat{\theta}_l) l^j} \bmod N.$$

As $\theta_l, \hat{\theta}_l < L$, $L < p$, $L < q$, $N = pq$ and p, q are primes, $GCD(\theta_l - \hat{\theta}_l, N) = 1$ and $(\theta_l - \hat{\theta}_l)^{-1} \bmod N$ exists. So

$$C_l = \prod_{j=0}^{t-1} F_j^{l^j} \bmod N.$$

Note that l can be any integer in $\{1,2,\ldots,n\}$. Therefore,

$$C_i = \prod_{j=0}^{t-1} F_j^{i^j} \bmod N \ \ for\ i = 1,2,\ldots,n.$$

□

6. COMPARISON AND CONCLUSION

Properties of the new PVSS scheme and the existing PVSS schemes are compared in Table 1 where K and $d_H(\log_2 e_i)$ have been defined in Section 2. Our comparison of computation focuses on the efficiency bottleneck in PVSS, public proof, and the verification of the validity of the encrypted shares. It also counts the number of exponentiations in large cyclic groups. The other operations like share generation, share encryption, share decryption by the share holders, secret reconstruction, and commitment to the share-generating polynomial are usually less costly and have a similar cost for different PVSS schemes, except that the two PVSS protocols with

Table 1. Comparison of PVSS Schemes

PVSS	Generality	Range proof	Computation in proof & verification
[8]	Yes	n	$\sum_{i=1}^{n}(t + 8d_H(\log_2 e_i))$
The first protocol in [13]	Yes	0	$n(t + 7K)$
The second protocol in [13]	Yes	0	$n(t + 7K)$
[3] with fast recovery	Yes	n	$\sum_{i=1}^{n}(t + 8d_H(\log_2 e_i))$
[3] with delayed recovery	No[1]	n	$n(t + 7)$
[10]	No[1]	0	$n(t + 6)$
[9]	Conditional[1]	n	$n(t + 26)$
New	Yes	0	$16n + t$

delayed recovery in [3] are inefficient in share decryption.

The comparison demonstrates the advantages of the new PVSS scheme proposed in this paper. It is a general PVSS solution without any limitations and is suitable for any application. It employs an ElGamal encryption so that all of the shareholders' encryption algorithms can employ the same decryption modulus to avoid range proofs. It is efficient as the first PVSS scheme to reduce the computational cost to $O(n)$.

REFERENCES

[1] "*The NIST special publication on computer security"* (sp 800-78 rev 1 of august 2007). 2007. Available at http://csrc.nist.gov/publications/nistpubs/.

[2] D Boneh and H Shacham. "*Fast variants of RSA. In CryptoBytes"*, Vol.5, No.1. 2002, pp.1-9.

[3] F Boudot and J Traore. "*Efficient public verifiable secret sharing schemes with fast or delayed recovery."* In *ICICS* '99, pp.87-102.

[4] F Boudot. "*Efficient proofs that a committed number lies in an interval*." In *EUROCRYPT* '00, LNCS1807, pp.431-444.

[5] D Chaum and T Pedersen. "*Wallet databases with observers."* In *CRYPTO* '92, LNCS740, pp.89-105.

[6] D Coppersmith. *"Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities."* In Journal of *Cryptology* Vol.10, No.4, 1997, pp.233-260.

[7] P Feldman. "*A practical scheme for non-interactive verifiable secret sharing*." In *FOCS* '87, pp.427-437.

[8] E Fujisaki and T Okamoto. "*A practical and provably secure scheme for publicly verifiable secret sharing and its applications*." In *EUROCRYPT* '98, pp.32-46.

[9] K Peng and F Bao. "*Efficient publicly verifiable secret sharing with correctness, soundness and ZK privacy*." In *WISA* '09, LNCS5932, pp.118-132.

[10] B Schoenmakers. "*A simple publicly verifiable secret sharing scheme and its application to electronic voting."* In *CRYPTO* '99, pp.149-164.

[11] C Schnorr. "*Efficient signature generation by smart cards*." Journal of *Cryptology*, 4, 1991, pp.161-174.

[12] A Shamir. "*How to share a secre*t." Communication of the *ACM*, 22(11):612-613, November 1979.

[13] M Stadler. "*Publicly verifiable secret sharing*." In *EUROCRYPT* '96, pp.190-199.

KUN PENG

Dr. Kun Peng received his Bachelor's degree in Software Engineering and his Master degree's in Computer Security from Huazhong University of Science and Technology in China. He obtained his PhD in information security from the Information Security Institute at the Queensland University of Technology in Australia in 2004. His main research interest is in applied public key cryptology. His main research interests include applied cryptology, network security, and secure e-commerce, and e-government. He is now a scientist at the Institute for Infocomm Research in Singapore.