Two Optimum Secret Sharing Schemes Revisited

Zhengjun Cao Olivier Markowitch
Department of Computer Sciences, Université Libre de Bruxelles, Belgium
Department of Mathematics, Shanghai University, China,
caoamss@gmail.com

Abstract

In 2006, Obana et al proposed two optimum secret sharing schemes secure against cheating. They extend the secret s in the Shamir's scheme to an array of three elements, (s, e_0, e_1), and construct two equations for checking validity. Each item in the equations should be reconstructed using Lagrange's interpolation. In this paper, we revisit these schemes by introducing a public hash function to construct equations for checking validity. The revisited schemes become more efficient because they only extend the secret to an array of two elements. The new scheme for a single secret saves about 1/3 cost of the original.

1 Introduction

Secret sharing was invented by both A. Shamir [12] and G. Blakley [1] independently in 1979. It refers to any method for distributing a secret amongst a group of participants, each of which is allocated a share of the secret. The secret can only be reconstructed when the shares are combined together; individual shares are of no use on their own. Secret sharing is a fundamental building block for many cryptographic protocols and is often used in the general composition of secure multiparty computations. In recent, secret sharing has still been an active research area because of its importance in cryptography [2-10,13-14].

In Shamir's (k, n) threshold secret sharing scheme, even a single user can fool other participants by submitting invalid shares at the secret reconstruction phase. Ogata et al [9] have presented an efficient scheme for detecting cheating. The scheme is proven to be secure only if the secret is uniformly distributed. In addition, the size of the secret in the scheme restricts possible value for the successful cheating probability.

In Asiacrypt'2006, S. Obana and T. Araki [8] introduced a method which uses universal hash functions to detect cheating and proposed two secret sharing schemes that employ the functions. The first scheme is nearly optimum with respect to the size of shares, namely, the size of shares is only one bit longer than its existing lower bound. The second scheme possesses a particular merit in that the parameter for the probability of successful cheating can be chosen without regard to the size of the secret. Further, the proposed schemes are proven to be secure regardless of the probability distribution of the secret.

They extend the secret key s in the Shamir's scheme to an array of three elements, (s, e_0, e_1), and introduce some accessorial data to resist cheating. They construct two equations for checking validity. Each item in the equations should be reconstructed using Lagrange's interpolation. In this paper, we revisit this schemes using a public hash function to construct equations for checking validity so that the new schemes become more efficient. Our schemes only extend the secret key to an array of two elements. The revisited scheme for a single secret saves about 1/3 cost of the original.

2 Preliminary

Lagrange interpolation Given a set of $k+1$ data points $(x_0, y_0), \cdots, (x_k, y_k)$, where no two x_j are the same, the interpolation polynomial in the Lagrange form is a linear combination

$$L(x) = \sum_{j=o}^{k} y_j \ell_j(x)$$

of Lagrange basis polynomials

$$\ell_j(x) = \prod_{i=0, i\neq j}^{k} \frac{x - x_i}{x_j - x_i}$$

Shamir's (k, n) secret sharing Suppose we want to use (k, n) threshold scheme to share our secret $s \in Z_p$, where p is a large number. Choose at random $(k-1)$ coefficients $a_1, a_2, \cdots, a_{k-1}$ and let $a_0 = s$. Build polynomial $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$. Let us construct any n points out of it, for instance set $i = 1, \cdots, n$ to retrieve $(i, f(i))$. Every participant is given a point (a pair of input

978-0-7695-3480-0/08 $25.00 © 2008 IEEE
DOI 10.1109/FITME.2008.95

to the polynomial and output). Given any subset of k of these pairs, we can find the coefficients of the polynomial by polynomial curve fitting, and then evaluate a_0, which is the secret.

Secret sharing schemes secure against cheating A secret sharing scheme capable of detecting cheating was first presented by Tompa and Woll [13]. They considered the scenario in which cheaters who do not belong to the access structure submit forged shares in the secret reconstruction phase. Such cheaters will succeed if another participants in the reconstruction accepts an incorrect secret. As in ordinary secret sharing schemes, each of these models consists of two algorithms. A share generation algorithm is the same as that in the ordinary secret sharing schemes. A secret reconstruction algorithm is slightly changed: it takes a list of shares as input, and outputs either a secret or a failed symbol.

3 Review of Obana-Araki secret sharing schemes

In Asiacrypt'2006, Obana and Araki [8] proposed two secret sharing schemes. They extend the Shamir's secret sharing scheme to an array of three elements and introduce some accessorial data to resist cheating. The underlying idea of the schemes is to use almost strongly universal hash functions for cheating detection. Under the circumstance, a randomly chosen key e is shared as well as the secret s using the Shamir's (k, n) threshold secret sharing scheme. These schemes can be described as follows.

3.1 A single secret case

[Share Generation] On input a secret $s \in GF(p)$, it outputs a list of shares $(v_1, \cdots, v_n)$ as follows:

1. Choose random $e_0, e_1 \in GF(p)$ such that $e_0 - se_1 = 0$.
2. Generate random polynomials $f_s(x), f_{e_0}(x), f_{e_1}(x) \in GF(p)[X]$ of degree $k-1$ such that $f_s(0) = s, f_{e_0}(0) = e_0$ and $f_{e_1}(0) = e_1$.
3. Compute $v_i = (f_s(i), f_{e_0}(i), f_{e_1}(i))$ and output $(v_1, \cdots, v_n)$.

[Secret reconstruction and validity check] On input a list of k shares $(v_{i1}, \cdots, v_{ik})$, it outputs a secret s as follows:

1. Reconstruct $\hat{s}, \hat{e}_0$ and $\hat{e}_1$ from $v_{i1}, \cdots, v_{ik}$ using Lagrange interpolation.
2. Output $\hat{s}$ if $\hat{e}_0 - \hat{s}\hat{e}_1 = 0$ holds.

3.2 Multiple secrets case

[Share Generation] On input a secret $s = (s_1, \cdots, s_N) \in GF(p)^N$, the share generation algorithm outputs a list of shares $(v_1, \cdots, v_n)$ according to the following procedure.

1. Choose random $e_0, e_1 \in GF(p)$ such that $e_0 - \sum_{j=1}^{N} s_j e_1^j = 0$.
2. Generate random polynomials of degree $k-1$, $f_s(x) \in GF(p^N)[X], f_{e_0}(x), f_{e_1}(x) \in GF(p)[X]$, such that
$$f_s(0) = s, f_{e_0}(0) = e_0, f_{e_1}(0) = e_1$$
3. Compute $v_i = (f_s(i), f_{e_0}(i), f_{e_1}(i))$ and output $(v_1, \cdots, v_n)$.

[Secret reconstruction and validity check] On input a list of k shares $(v_{i1}, \cdots, v_{ik})$, it outputs a secret s as follows:

1. Reconstruct $\hat{s}, \hat{e}_0, \hat{e}_1$ from $v_{i1}, \cdots, v_{ik}$.
2. Output $\hat{s}$ if $\hat{e}_0 - \sum_{j=1}^{N} \hat{s}_j \hat{e}_1^j = 0$ holds.

4 Analysis of Obana-Araki secret sharing schemes

In Obana-Araki secret sharing schemes, the authors extend $f_s(i)$ to
$$v_i = (f_s(i), f_{e_0}(i), f_{e_1}(i))$$
against cheaters.

For validity check, they construct equations
$$\hat{e}_0 - \hat{s} \cdot \hat{e}_1 = 0 \quad \text{(a single key)}$$
$$\hat{e}_0 - \sum_{j=1}^{N} \hat{s}_j \hat{e}_1^j = 0 \quad \text{(multiple keys)}$$

Notice that the main cost almost arises from the reconstructions of $\hat{s}, \hat{e}_0, \hat{e}_1$. To cut cost, intuitively, we should reduce the number of reconstructions. Thus it's better to extend the $f_s(i)$ to an array of only tow elements.

Remark: A cursoriness in the description "choose random $e_0, e_1 \in GF(p)$ such that $e_0 - \sum_{j=1}^{N} s_j e_1^j = 0$" should be corrected. Actually, it is very difficult to choose two random number $e_0, e_1 \in GF(p)$ given $s = (s_1, \cdots, s_N) \in GF(p)^N$ such that the above equation holds. It's better to say "choose random $e_1 \in GF(p)$ and compute e_0 such that $e_0 - \sum_{j=1}^{N} s_j e_1^j = 0$". Under the circumstance, the randomness of e_0 does not matter about the scheme's security.

5 Obana-Araki secret sharing schemes revisited

In this section, we will extend the $f_s(i)$ to an array of only tow elements and construct equations for checking validity. It's well known that a general method for checking validity is to introduce a public hash function H and check the consistency of the fingerprint. Now we describe the revisited schemes as follows.

5.1 A single secret case

[Share Generation] On input a secret $s \in GF(p)$ and a public hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, it outputs a list of shares $(v_1, \cdots, v_n)$ as follows:

1. Compute $e = H(s)$.
2. Generate random polynomials

$$f_s(x), \quad f_e(x) \in GF(p)[X]$$

of degree $k-1$ such that $f_s(0) = s, f_e(0) = e$.

3. Compute $v_i = (f_s(i), f_e(i))$ and output $(v_1, \cdots, v_n)$.

[Secret reconstruction and validity check] On input a list of k shares $(v_{i1}, \cdots, v_{ik})$, it outputs a secret s as follows:

1. Reconstruct $\hat{s}, \hat{e}$ from $v_{i1}, \cdots, v_{ik}$ using Lagrange interpolation.
2. Output $\hat{s}$ if $\hat{e} = H(\hat{s})$ holds.

5.2 Multiple secrets case

[Share Generation] On input a secret $s = (s_1, \cdots, s_N) \in GF(p)^N$ and a public hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, the share generation algorithm outputs a list of shares $(v_1, \cdots, v_n)$ according to the following procedure.

1. Compute $e = H(s_1 \,||\, \cdots ||s_N)$
2. Generate random polynomials

$$f_s(x) \in GF(p^N)[X], \quad f_e(x) \in GF(p)[X]$$

of degree $k-1$ such that $f_s(0) = s, f_e(0) = e$.

3. Compute $v_i = (f_s(i), f_e(i))$ and output $(v_1, \cdots, v_n)$.

[Secret reconstruction and validity check] On input a list of k shares $(v_{i1}, \cdots, v_{ik})$, it outputs a secret s as follows:

1. Reconstruct $\hat{s}, \hat{e}$ from $v_{i1}, \cdots, v_{ik}$.
2. Output $\hat{s}$ if $\hat{e} = H(\hat{s}_1 \,||\, \cdots ||\hat{s}_N)$ holds.

5.3 Security

For simplicity, we only analyze the first scheme informally.

If a malicious participant wants to fool others, he has to play a trick in the phase of secret reconstruction and validity check. Comparing the original equation for checking validity and the new equation, namely,

$$e_0 - se_1 = 0 \quad \text{and} \quad e = H(s)$$

the secret s is bound to the random number e_1 in the former, whereas it is bound to its fingerprint in the latter. Without doubt, the intractability of the latter is hard than that of the former, according to general assumptions of a cryptographic hash function.

As for the amount of leaked information, the array of three elements $(f_s(i), f_{e_0}(i), f_{e_1}(i))$ is definitely greater than the array of two elements $(f_s(i), f_e(i))$. Therefore, the new scheme is more robust than the original.

5.4 efficiency

It's clear that the main cost almost arises from the reconstructions of $\hat{s}, \hat{e}_0, \hat{e}_1$ in the original schemes.

For a single secret case, the revisited scheme saves about 1/3 cost since the hash function usually runs greatly faster than Lagrange's interpolation for recovering a key .

As for the other, the simplified scheme saves about $\frac{1}{N+2}$ cost of the original if the cost of reconstruction of s_i is viewed as same as that of reconstruction of e, where N is the number of multiple secrets.

In addition, the storage of the hash function H can also be saved since there are a lot of public hash functions available in daily life.

6 Conclusion

Cryptographic hash functions play a fundamental role in modern cryptography. Hash functions take a message as input and produce an output referred to fingerprint. It's prevalent in cryptography to use hash functions to provide message authentication and data integrity [11]. Hence, it is feasible to introduce a hash function in the original schemes.

7 Acknowledgement

This work is supported by National Natural Science Foundation of China (60873227) and Science and Technology Innovation Foundation of Shanghai.

References

[1] G.Blakley. Safeguading cryptographic keys, In: *Proceedings of the National Computer Conference*, 1979, pp. 313-317

[2] R.Cramer, I.Damgard, U.Maurer. General Secure Multi-party Computation from any Linear Secret-Sharing Scheme, In: *Advances in cryptology-Asiacrypt'Eurocrypt'2000*, Lecture notes in computer science 1807, Springer, 2000, pp. 316-334

[3] L. Czirimaz. The size of a share must be large, Journal of Cryptology 10 (1997), pp. 223-231.

[4] C.Ding, T.Laihonen, A.Renvall. Linear multi-secret sharing schemes and errorcorrecting codes, *Journal of Universal Computer Science*, Vol.3, No.9, 1997, pp. 1023-1036

[5] K.Kurosawa, S.Obana, W.Ogata. t-Cheater Identifiable (k, n) Secret Sharing Schemes, In: *Advances in cryptology-Asiacrypt'Crypto'1995*, Lecture notes in computer science 963, Springer, 1995, pp. 410-423

[6] K.Kulesza, Z.Kotulski, J.Pieprzyk. On Alternative Approach for Verifiable Secret Sharing, http://iacr.org/2003/035

[7] M.Liu, L.Xiao, Z.Zhang. Multiplicative Linear Secret Sharing Schemes Based on Connectivity of Graphs. *IEEE Transactions on Information Theory*, 2007, 53(11): 3973-3978

[8] S.Obana, T.Araki. Almost Optimum Secret Sharing Schemes Secure Against Cheating for Arbitrary Secret Distribution, In: *Advances in cryptology-Asiacrypt'2006*, Lecture notes in computer science 4284, Springer, 2006,pp. 364-379

[9] W.Ogata, K.Kurosawa,D.Stinson. Optimum Secret Sharing Scheme Secure against Cheating, *SIAM Journal on Discrete Mathematics*, Vol. 20, No. 1, pp. 79-95

[10] T.Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing, In: *Advances in cryptology-Asiacrypt'Crypto'1991*, Lecture notes in computer science 576, Springer, 1991, pp. 129-149

[11] C.Schnorr, Efficient signature generation for smart card, *Journal of Cryptology*, Vol 4, No 03, 1991, pp. 161-174

[12] A.Shamir. How to Share a Secret, *Communications of the ACM*, Vol. 22, No. 11, 1979, pp. 612-613

[13] M.Tompa, H.Woll. How to Share a Secret with Cheaters, *Journal of Cryptology*, Vol. 1, No. 3, 1989, pp. 133-138

[14] Z.Zhang, M.Liu, L.Xiao. Parallel Multi-party Computation from Linear Multi-secret Sharing Schemes, In: *Advances in cryptology-Asiacrypt'2005*, Lecture notes in computer science 3788, Springer, 2005, pp. 156-173