Sharing DSS by the Chinese Remainder Theorem

Kamer Kaya, Ali Aydın Selçuk

Department of Computer Engineering
Bilkent University
Ankara, 06800, Turkey
{kamer,selcuk}@cs.bilkent.edu.tr

November 16, 2008

Abstract

A new threshold scheme for the Digital Signature Standard is proposed using Asmuth-Bloom secret sharing based on the Chinese Remainder Theorem. The proposed scheme is simple and can be used practically in real life.

Keywords: Asmuth-Bloom secret sharing, threshold cryptography, function sharing, DSS.

1 Introduction

Threshold cryptography deals with the problem of sharing a highly sensitive secret among a group of n users so that only when a sufficient number t of them come together can the secret be reconstructed. This problem is known as the secret sharing problem and several secret sharing schemes (SSS) have been proposed in the literature (e.g., [1, 2, 8]).

Another problem threshold cryptography deals with is the function sharing problem. A function sharing scheme (FSS) requires distributing the function's computation according to the underlying SSS such that each part of the computation can be carried out by a different user and then the partial results can be combined to yield the function's value without disclosing individual secrets. All FSSs in the literature, e.g., [3, 4, 7, 9], proposed for various cryptosystems have traditionally used Shamir's SSS [8] until a recent work by Kaya and Selçuk [6] which showed how to use the Asmuth-Bloom SSS (AB-SSS [1] for function sharing.

The Digital Signature Standard (DSS) is the current U.S. standard for digital signatures. Sharing DSS is an interesting problem and a neat solution was given by Gennaro et al. [5] based on Shamir's SSS. Here we give an alternative solution for this problem based on a modified version of the Asmuth-Bloom SSS (AB-SSS).

The rest of the paper is organized as follows: In Section 2 and 3, we describe the Digital Signature Standard and the Asmuth-Bloom secret sharing scheme, respectively. In Section 4, the threshold DSS scheme based on the Asmuth-Bloom SSS is proposed. Section 5 concludes the paper.

2 Digital Signature Standard:

The DSS is summarized below:

- *Key Generation Phase:* Let p and q be large prime numbers where $q|p-1$ and $g \in \mathbb{Z}_p^*$ be an element of order q. The private key $\alpha \in_R \mathbb{Z}_q^*$ is chosen randomly and the public key $\beta = g^\alpha \bmod p$ is computed.
- *Signing Phase:* The signer first chooses a random ephemeral key $k \in_R \mathbb{Z}_q^*$ and then computes the signature (r, s) where

$$r = (g^{k^{-1}} \bmod p) \bmod q$$
$$s = k(w + \alpha r) \bmod q$$

  for a hashed message $w \in \mathbb{Z}_q$.
- *Verification Phase:* The signature (r, s) is verified by checking

$$r \stackrel{?}{=} (g^{ws^{-1}} \beta^{rs^{-1}} \bmod p) \bmod q$$

  where s^{-1} is computed in $\mathbb{Z}_q^*$.

3 Asmuth-Bloom Secret Sharing Scheme

The phases of the Asmuth-Bloom SSS are described below:

- *Dealer Phase:* Let d be the secret to be shared, n be the number of users, and t be the threshold value. Let $m_0 < m_1 < m_2 < \ldots < m_n$ be relatively prime integers such that $d < m_0$ and

$$m_0{}^2 \prod_{i=1}^{t-1} m_{n-i+1} < \prod_{i=1}^{t} m_i$$

  (see [6] for a detailed discussion). Let M denote $\prod_{i=1}^{t} m_i$. The dealer computes $y = d + Am_0$ where A is a random positive integer such that $y < M$. The share of the ith user is $y_i = y \bmod m_i$.

- *Combiner Phase:* Let S be a coalition of t users gathered to construct the secret. Let M_S denote $\prod_{i \in S} m_i$.
  - Let $M_{S\setminus\{i\}}$ denote $\prod_{j\in S, j\neq i} m_j$ and $M'_{S,i}$ be the multiplicative inverse of $M_{S\setminus\{i\}}$ in $\mathbb{Z}_{m_i}$, i.e., $M_{S\setminus\{i\}} M'_{S,i} \equiv 1 \pmod{m_i}$. First, the ith user computes

$$u_i = y_i M'_{S,i} M_{S\setminus\{i\}} \bmod M_S.$$

  - The users first compute

$$y = \left(\sum_{i \in S} u_i \right) \bmod M_S$$

and then obtain the secret d by computing

$$d = y \bmod m_0.$$

We will use the notation $d \overset{t}{\leftrightarrow} (y_1, y_2, \ldots, y_n)$ to denote a (t, n)-SSS with secret d and shares $\{y_1, y_2, \ldots, y_n\}$.

3.1 Arithmetic Properties of the Asmuth-Bloom SSS

Suppose multiple secrets are shared with common parameters t, n, and moduli m_is. The shareholders can use the following properties to obtain new shares for the sum and product of the shared secrets.

Proposition 1 *Let $d_1, d_2, \cdots, d_\ell$ be secrets shared by AB-SSS with common parameters t, n, and moduli m_is, for some $\ell < m_0$. Let y_{ij} be the share of the ith user for secret d_j. Then, for $\overline{d} = (\sum_{i=1}^{\ell} d_i) \bmod m_0$ and $\overline{y}_i = (\sum_{j=1}^{\ell} y_{ij}) \bmod m_i$, we have $\overline{d} \overset{t+1}{\leftrightarrow} (\overline{y}_1, \overline{y}_2, \cdots, \overline{y}_n)$.*

Proof 2 *For $\overline{y} = \sum_{i=1}^{\ell}(d_i + A_i m_0)$, we have $\overline{y}_i \equiv \overline{y} \bmod m_i$. Note that $\overline{y} < \ell M < M_S$ for any coalition S where $|S| \geq t+1$. Hence, a coalition S of $t+1$ users can construct $\overline{y} \in M_S$ and obtain $\overline{d} = \overline{y} \bmod m_0$.*

Proposition 3 *Let d_1, d_2 be secrets shared by AB-SSS with common parameters t, n and moduli m_is. Let y_{ij} be the share of the ith user for secret d_j. Then, for $\overline{d} = d_1 d_2 \bmod m_0$ and $\overline{y}_i = y_1 y_2 \bmod m_i$, we have $\overline{d} \overset{2t}{\leftrightarrow} (\overline{y}_1, \overline{y}_2, \cdots, \overline{y}_n)$.*

Proof 4 *For $\overline{y} = \prod_{i=1}^{2}(d_i + A_i m_0)$, we have $\overline{y}_i \equiv \overline{y} \bmod m_i$. Note that $\overline{y} < M^2 < M_S$ for any coalition S where $|S| \geq 2t$. Hence, a coalition S of $2t$ users can construct $\overline{y} \in M_S$ and obtain $\overline{d} = \overline{y} \bmod m_0$.*

4 Sharing DSS

To obtain a threshold DSS scheme, first the dealer generates the private key α and shares it among the users by (t, n) AB-SSS with $m_0 = q$. Then a signing coalition S can sign a message in a threshold fashion without requiring a trusted party. Note that anyone can forge signatures if he knows k for a valid signature (r, s). Hence, $r = (g^{k^{-1}} \bmod p) \bmod q$ must be computed in a way that no one obtains k. Here, we first explain the necessary primitives that will be used to solve this problem and then describe the overall threshold signature scheme together. Below, S denotes the signing coalition of size $2t + 2$.

4.1 Joint Random Secret Sharing

In a joint random secret sharing (Joint-RSS) scheme, each user in the signing coalition S contributes something to the secret generation process and obtains a share for the resulting random secret as described below:

1. Each user $j \in S$ chooses a random secret $d_j \in \mathbb{Z}_{m_0}$ and shares it as $d_j \overset{t}{\leftrightarrow} (y_{1j}, y_{2j}, \cdots, y_{nj})$ where y_{ij} is the share of the ith user.

2. The ith user computes $\overline{y}_i = (\sum_{j=1}^{n} y_{ij}) \bmod m_i$. By Proposition 1, $\overline{d} \overset{t+1}{\leftrightarrow} (\overline{y}_1, \overline{y}_2, \ldots, \overline{y}_n)$ is a valid SSS for $\overline{d} = (\sum_{i=1}^{n} d_i) \bmod m_0$ assuming $n < m_0$.

4.2 Computing $g^d \bmod p$

In DSS, we need to share and compute $g^d \bmod p$ for a joint random secret d. Here we give a scheme, Joint-Exp-RSS, to construct an approximate value for $g^d \bmod p$. This approximate value will later be corrected through a separate correction process.

1. Use Joint-RSS to generate and share a secret d as $d \overset{t+1}{\leftrightarrow} (y_1, y_2, \ldots, y_n)$. Let $S' \subset S$ be a coalition of size $t + 1$ that wants to compute $f_d = g^d \bmod p$.

2. Each user $i \in S'$ computes $u_{i,d} = (y_i M_{S' \setminus \{i\}} M'_{S',i}) \bmod M_{S'}$ where $M'_{S',i}$ is the inverse of $M_{S' \setminus \{i\}} \bmod m_i$, and broadcasts $f_{i,d} = g^{u_{i,d}} \bmod p$.

3. The approximate value for $g^d \bmod p$ is computed as $f_{d'} = \prod_{i \in S'} f_{i,d} \bmod p$.

Observe that $d = ((\sum_{i \in S'} u_i) \bmod M_{S'}) \bmod q$ whereas this construction process computes $f_{d'} = g^{d'} \bmod p$ for $d' = \sum_{i \in S'} u_i \bmod q$. Since there are $t + 1$ users in S' and $u_i < M_{S'}$ for all i, $d \equiv d' - \delta_d M_{S'} \bmod q$ for some integer $0 \leq \delta_d \leq t$.

4.3 Computing $g^{k^{-1}} \bmod p$

In DSS, we need to compute $r = g^{k^{-1}} \bmod p$ in such a way that neither k nor k^{-1} is known by any user. The following JOINT-EXP-INVERSE procedure computes r without revealing k:

1. S uses JOINT-RSS to jointly share random secrets $k \overset{t+1}{\leftrightarrow} (k_1, k_2, \ldots, k_n)$ and $a \overset{t+1}{\leftrightarrow} (a_1, a_2, \ldots, a_n)$ and constructs $v = ak$ from shares $v_i = a_i k_i \bmod m_i$, $i \in S$. Note that $v \overset{2t+2}{\leftrightarrow} (v_1, v_2, \ldots, v_n)$ by Proposition 3.

2. To compute $g^a \bmod q$, each user $i \in S'$ computes
$$u_{i,a} = (a_i M_{S' \setminus \{i\}} M'_{S',i}) \bmod M_{S'}$$
and broadcasts $f_{i,a} = g^{u_{i,a}} \bmod p$. The approximate value is computed as
$$f_{a'} = \prod_{i \in S'} f_{i,a} \bmod p = g^{a'} \bmod p$$
for some $a' = a + \delta_a M_{S'}$, $0 \leq \delta_a \leq t$. S' corrects $f_{a'}$ through the following correction procedure:

   (a) Let $S' \subset S$ be a set of $t+1$ users. Each user $i \in S'$ computes
   $$u_{i,k} = (k_i M_{S' \setminus \{i\}} {M'}_{S',i}) \bmod M_{S'}$$
   and broadcasts $f_{i,k} = g^{u_{i,k}} \bmod p$ and $f_{i,ak} = {f_{a'}}^{u_{i,k}} \bmod p$. After that,
   $$f_{k'} = \prod_{i \in S'} f_{i,k} \bmod p = g^{k'} \bmod p,$$
   $$f_{a'k'} = \prod_{i \in S'} f_{i,ak} \bmod p = g^{a'k'} \bmod p$$
   are computed, where $k' = k + \delta_k M_{S'}$ for some $0 \leq \delta_k \leq t$. Note that
   $$\begin{aligned} f_{a'k'} &= g^{ak + a\delta_k M_{S'} + k\delta_a M_S'' + \delta_a \delta_k M_{S'}^2} \bmod p \\ &= g^v (f_{a'} g^{-\delta_a M_{S'}})^{\delta_k M_{S'}} (f_{k'} g^{-\delta_k M_{S'}})^{\delta_a M_{S'}} g^{\delta_a \delta_k M_{S'}^2} \bmod p \\ &= g^v f_{a'}^{\delta_k M_{S'}} f_{k'}^{\delta_a M_{S'}} g^{-\delta_a \delta_k M_{S'}^2} \bmod p \end{aligned}$$

   (b) S' checks the following equality for all $0 \leq j_a, j_k \leq t$
   $$f_{a'k'} \overset{?}{=} g^v f_{a'}^{j_k M_{S'}} f_{k'}^{j_a M_{S'}} g^{-j_a j_k M_{S'}^2} \bmod p \tag{1}$$
   and finds the $(j_a = \delta_a, j_k = \delta_k)$ pair that satisfies this equality. Once δ_a is found $f_a = g^a \bmod p = f_{a'} g^{-\delta_a M_{S'}} \bmod p$ can be computed.

3. The signing coalition S computes $g^{k^{-1}} \bmod p = {f_a}^{(v^{-1})} \bmod p$.

The (j_a, j_k) pair, $0 \leq j_a, j_k \leq t$, found for (1) is unique with overwhelming probability given that $(t+1)^2 \ll q$.

4.4 Threshold DSS Scheme

The phases of the proposed threshold DSS scheme are as follows:

- *Key Generation Phase:* Let $\alpha \in_R \mathbb{Z}_q^*$ be the private signature key. The dealer sets $m_0 = q$ and shares $\alpha \stackrel{t}{\leftrightarrow} (\alpha_1, \alpha_2, \ldots, \alpha_n)$.

- *Signing Phase:* To sign a hashed message $w \in \mathbb{Z}_q$, the signing coalition S of size $2t+2$ first computes $r = (g^{k^{-1}} \bmod p) \bmod q$ by JOINT-EXP-INVERSE. To compute $s = k(w + r\alpha) \bmod q$, each user $i \in S$ computes

$$s_i = k_i(w + r\alpha_i) \bmod m_i.$$

  Since α is shared (t, n), the value $y = \alpha + A_\alpha m_0$ is less than M. Hence, $w + ry < m_0 + m_0 y < (m_0 + 1)M$ and a coalition of size $t+1$ is sufficient to compute $w + ry$ and obtain $w + r\alpha \bmod q$. Since the threshold for secret k is also $t+1$, by Proposition 3, $s \stackrel{2t+2}{\leftrightarrow} (s_1, s_2, \ldots, s_n)$ and s is computed by $2t+2$ partial signatures.

- *Verification Phase* is the same as the standard DSS verification.

5 Conclusion

In this paper, we investigated how to share the signing function used in the Digital Signature Standard by using the Asmuth-Bloom secret sharing scheme. We proposed a t-out-of-n threshold signature scheme based on the Chinese Remainder Theorem.

References

[1] C. Asmuth and J. Bloom. A modular approach to key safeguarding. *IEEE Trans. Information Theory*, 29(2):208–210, 1983.

[2] G. Blakley. Safeguarding cryptographic keys. In *Proc. of AFIPS National Computer Conference*, 1979.

[3] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Proc. of CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990.

[4] Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures. In *Proc. of CRYPTO'91*, volume 576 of *LNCS*, pages 457–469. Springer-Verlag, 1992.

[5] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. *Information and Computation*, 164(1):54–84, 2001.

[6] K. Kaya and A. A. Selçuk. Threshold cryptography based on Asmuth-Bloom secret sharing. *Information Sciences*, 177(19):4148–4160, 2007.

[7] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. How to share a function securely? In *Proc. of STOC94*, pages 522–533, 1994.

[8] A. Shamir. How to share a secret? *Comm. ACM*, 22(11):612–613, 1979.

[9] V. Shoup. Practical threshold signatures. In *Proc. of EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 207–220. Springer-Verlag, 2000.