Threshold Cryptography Based on Blakley Secret Sharing

İlker Nadi Bozkurt, Kamer Kaya, Ali Aydın Selçuk
Department of Computer Engineering
Bilkent University
Ankara, 06800, Turkey
Email: {bozkurti,kamer,selcuk}@cs.bilkent.edu.tr

Ahmet M. Güloğlu
Department of Mathematics
Bilkent University
Ankara, 06800, Turkey
Email: guloglua@fen.bilkent.edu.tr

***Abstract*—Function sharing deals with the problem of distribution of the computation of a function (such as decryption or signature) among several parties. The necessary values for the computation are distributed to the participating parties using a secret sharing scheme (SSS). Several function sharing schemes have been proposed in the literature, with most of them using Shamir secret sharing as the underlying SSS. In this paper, we investigate how threshold cryptography can be conducted with Blakley secret sharing scheme and present a novel function sharing scheme for the RSA cryptosystem. The challenge is that constructing the secret in Blakley's SSS requires the solution of a linear system which normally involves computing inverses, while computing inverses modulo $\phi(N)$ cannot be tolerated in a threshold RSA system in any way.**

***Index Terms*—threshold cryptography, RSA, function sharing, Blakley secret sharing.**

I. Introduction

The secure storage of the private keys of a cryptosystem is an important problem. The possession of a highly sensitive key by an individual may not be desirable as the key can easily be lost or as the individual may not be fully trusted. Giving copies of the key to more than one individual increases the risk of compromise. A solution to this problem is to give shares of the key to several individuals, forcing them to cooperate to find the secret key. This not only reduces the risk of losing the key but also makes compromising the key more difficult. In threshold cryptography, secret sharing deals with this problem, namely, 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. Well-known secret sharing schemes (SSS) in the literature include Shamir [12] based on polynomial interpolation, Blakley [2] based on hyperplane geometry, and Asmuth-Bloom [1] based on the Chinese Remainder Theorem.

A shortcoming of secret sharing schemes is the need to reveal the secret shares during the reconstruction phase. The system would be more secure if the subject function can be computed without revealing the secret shares or reconstructing the secret. This is known as the function sharing problem. A function sharing scheme 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 the individual secrets. Several protocols for function sharing have been proposed in the literature [3], [4], [5], [6], [13], [8], [11]. Nearly all the existing solutions for function sharing uses Shamir secret sharing as the underlying SSS.

In this paper we present a novel threshold RSA signature scheme based on Blakley's secret sharing as we explain below.

A. Secret Sharing Schemes

The problem of secret sharing and the first solutions were introduced in 1979 independently by Shamir [12] and Blakley [2]. A (t, n)-secret sharing scheme is used to distribute a secret d among n people such that any coalition of size t or more can construct d but smaller coalitions cannot.

Shamir secret sharing is based on polynomial interpolation over a finite field. It uses the fact that we can find a polynomial of degree $t-1$ given t data points. To generate a polynomial $f(x) = \sum_{i=0}^{t-1} a_i x^i$, a_0 is set to the secret value and the coefficients a_1 to a_{t-1} are assigned random values in the field. The value $f(i)$ is given to user i. When t out of n users come together, they can construct the polynomial using Lagrange interpolation and can find the secret.

Blakley secret sharing scheme has a different approach based on hyperplane geometry: To implement a (t, n) threshold scheme, each of the n users is given a hyperplane equation in a t dimensional space over a finite field such that each hyperplane passes through a certain point. The intersection point of the hyperplanes is the secret. When t users come together, they can solve the system of equations to find the secret.

B. Function Sharing Schemes

Function sharing is the concept of distribution of the computation of a function such that when a sufficient number of users come together they can compute the value of the function without revealing their secret shares but less than the threshold number of users cannot. This problem is related to secret sharing as the secret values needed for partial computations are distributed using secret sharing.

Several solutions for sharing the RSA, ElGamal, and Paillier private key operations have been proposed in the literature [3], [4], [5], [6], [7], [10], [11], [13], [9]. Almost all of these schemes have been based on the Shamir SSS.

The additive nature of the Lagrange's interpolation formula used in the combining phase of Shamir's scheme makes it an attractive choice for function sharing, but it also provides several challenges. One of the most significant challenges is the computation of inverses in $\mathbb{Z}_{\phi(N)}$ for the division operations in Lagrange's formula where $\phi(N)$ should not be known by the users. There are two main difficulties in this respect:

1) An inverse x^{-1} will not exist modulo $\phi(N)$ if $\gcd(x, N) \neq 1$.
2) Even when x^{-1} exists it should not be computable by a user, since that would enable computing $\phi(N)$.

Early solutions to this problem, albeit not very efficient, were given in [3], [11]. Afterwards an ingenious solution was given by Shoup [13] where he removed the need of taking inverses in Lagrange interpolation.

Shoup's practical RSA scheme has inspired similar works on different cryptosystems. Fouque et al. [7] proposed a similar threshold solution for the Paillier cryptosystem and used it in e-voting and lottery protocols. Later, Lysyanskaya and Peikert [10] improved this work and obtained a threshold Paillier encryption scheme secure under the adaptive security model. The current paper is also inspired by Shoup's work.

C. Our Contribution

In this work, we show how to do threshold RSA signatures using Blakley SSS. Blakley's scheme, which is based on solving linear systems, naturally requires computing inverses for reconstructing the secret. We show, in a spirit similar to Shoup's work, how to utilize Blakley's SSS for threshold cryptography while avoiding computation of inverses modulo $\phi(N)$ completely.

II. Blakley's Secret Sharing Scheme

Blakley's SSS uses hyperplane geometry to solve the secret sharing problem. The secret is a point in a t-dimensional space and n shares are affine hyperplanes that pass through this point. An affine hyperplane in a t-dimensional space with coordinates in a field $\mathcal{F}$ can be described by a linear equation of the following form:

$$a_1x_1 + a_2x_2 + \ldots + a_tx_t = b.$$

The intersection point is obtained by finding the intersection of any t of these hyperplanes. The secret can be any of the coordinates of the intersection point or any function of the coordinates. We take the secret to be the first coordinate of the point of intersection.

A. Dealing Phase

Let m be a prime and let $\mathcal{F} = \mathbb{Z}_{\Updownarrow}$ be the field we are working on. The dealer generates a secret point x in $\mathcal{F}^{\sqcup}$, where the first coordinate $x[1]$ is set to the secret value (the RSA private key d in our case) and sets the values of the other coordinates randomly from the field $\mathcal{F}$. The ith user will get a hyperplane equation over $\mathcal{F}$,

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{it}x_t = y_i. \tag{1}$$

For a (t, n) threshold scheme there will be n such hyperplane equations, and hence we will have an $n \times t$ linear system,

$$Ax = y. \tag{2}$$

The dealer then sends the secret value of y_i along with $a_{i1}, \ldots, a_{it}$ to user i. The coefficients a_{ij} are not sensitive and can be made public if needed.

B. Share Combining Phase

Share combining step is simply finding the solution of a linear system of equations. Suppose that a coalition $\mathcal{S} = \{i_1, \ldots, i_t\}$ of users come together. They form a matrix $A_{\mathcal{S}}$ using their hyperplane equations and solve

$$A_{\mathcal{S}}x = y_{\mathcal{S}}, \tag{3}$$

where $y_{\mathcal{S}}$ is the vector of the secret shares of the users. The secret is found as the first coordinate of the solution.

III. Sharing RSA Signature Computation

In this section, we describe our threshold RSA signature scheme with Blakley secret sharing.

A. Setup

In the RSA setup phase, choose two large primes p and q, and compute the RSA modulus as $N = pq$. The public key e is chosen as a prime number relatively prime to $\phi(N)$, the details of which will be explained in Section V. After choosing e, the private key d is computed such that $ed \equiv 1 \pmod{\phi(N)}$. Then the dealer shares the private key d among n users using Blakley SSS in $\mathbb{Z}_{\phi(N)}$.

B. Signing

Let $H(.)$ be a hash function mapping input messages to $\mathbb{Z}_N^*$ and let $w = H(M) \in \mathbb{Z}_N^*$ be the hashed message to be signed. Assume a coalition $\mathcal{S}$ of size t wants to obtain the signature $s = w^d \bmod N$.

1) Generating the Partial Signatures: Let $\mathcal{S} = \{i_1, \dots, i_t\}$ be the coalition of t users, forming the linear system

$$A_{\mathcal{S}} x = y_{\mathcal{S}}.$$

Let c_{ij} be the ij-th cofactor of matrix $A_{\mathcal{S}}$ and let $C_{\mathcal{S}}$ be the adjugate matrix,

$$C_{\mathcal{S}} = \begin{pmatrix} c_{11} & c_{21} & \cdots & c_{t1} \\ c_{12} & c_{22} & \cdots & c_{t2} \\ \vdots & \vdots & \ddots & \vdots \\ c_{1t} & c_{2t} & \cdots & c_{tt} \end{pmatrix}.$$

If we denote the determinant of $A_{\mathcal{S}}$ by $\Delta_{\mathcal{S}}$, we have

$$A_{\mathcal{S}} C_{\mathcal{S}} = C_{\mathcal{S}} A_{\mathcal{S}} = \Delta_{\mathcal{S}} I_t, \tag{4}$$

where I_t denotes the $t \times t$ identity matrix.

For our scheme, each user $i \in \mathcal{S}$ computes his partial signature as

$$s_i = w^{c_{i1} y_i} \bmod N. \tag{5}$$

2) Combining the Partial Signatures: To combine the partial signatures, we simply compute

$$s_p = s_{i_1} s_{i_2} \dots s_{i_t} \bmod N. \tag{6}$$

Note that, from equation (4), we have

$$s_p = w^{\Delta_{\mathcal{S}} d} \bmod N. \tag{7}$$

Given that e is a prime number relatively prime to $\Delta_{\mathcal{S}}$, it is easy to compute the signature $s = w^d \bmod N$ from s_p. Take

$$s = s_p^a w^b \bmod N, \tag{8}$$

where a and b are integers such that

$$\Delta_{\mathcal{S}} a + eb = 1, \tag{9}$$

which can be obtained by the extended Euclidean algorithm on $\Delta_{\mathcal{S}}$ and e.

IV. Solution of the Linear System

In Blakley's SSS, the private key is found by solution of the linear system $A_{\mathcal{S}} x = y_{\mathcal{S}}$. However, this system may not have a unique solution over $\mathbb{Z}_{\phi(N)}$. If $\gcd(\Delta_{\mathcal{S}}, \phi(N)) > 1$, the matrix $A_{\mathcal{S}}$ will not have an inverse modulo $\phi(N)$, and the linear system will have many different solutions. Interestingly, our threshold signature scheme computes the correct signature in this case as well.

When $\gcd(\Delta_{\mathcal{S}}, \phi(N)) > 1$ and the linear system yields many different solutions for d, note that the value $\Delta_{\mathcal{S}} d$ is a fixed number for all these possible solutions, and is equal to

$$\Delta_{\mathcal{S}} d = \Sigma_i c_{i1} y_i.$$

Hence, the incomplete signature

$$\begin{aligned} s_p &= w^{\Sigma_i c_{i1} y_i} \bmod N \\ &= w^{\Delta_{\mathcal{S}} d} \bmod N \end{aligned}$$

is the same for every solution of the system $A_{\mathcal{S}} x = y_{\mathcal{S}}$.

Then the signature s is obtained from s_p as

$$s = s_p^a w^b \bmod N,$$

where a and b are the integer solutions of $\Delta_{\mathcal{S}} a + eb = 1$. Hence, the signature s is $w^d \bmod N$ for the right d value, computed according to the public key e.

V. Choosing *e*

The choice of e is critical in the setup phase because the solution depends on e and $\Delta_{\mathcal{S}}$ being relatively prime. To achieve this, we can either choose a special matrix whose determinant we know to be relatively prime to e, or choose e as a sufficiently large prime according to t and n so that the probability that $\Delta_{\mathcal{S}}$ is divisible by e will be negligible for any coalition $\mathcal{S}$.

A. Choosing e Probabilistically

The probability of a random integer's being divisible by a prime e is $1/e$. So, if we have a (t, n) threshold scheme, the probability that the determinant of none of the $\binom{n}{t}$ $A_{\mathcal{S}}$ matrices will be divisible by e is $\left(1 - \frac{1}{e}\right)^{\binom{n}{t}}$. If we take $e \gg \binom{n}{t}$, we have

$$\left(1 - \frac{1}{e}\right)^{\binom{n}{t}} \approx 1 \ . \tag{10}$$

B. Using a Vandermonde Matrix

A simple choice for the matrix A that enables us to guarantee that e will be relatively prime to the determinant of the coefficient matrix is to choose the rows of the matrix A as the rows of a Vandermonde matrix. Then, $A_{\mathcal{S}}$ will have the following form for any coalition $\mathcal{S}$:

$$A_{\mathcal{S}} = \begin{pmatrix} 1 & a_1 & a_1^2 & \ldots & a_1^{t-1} \\ 1 & a_2 & a_2^2 & \ldots & a_2^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_t & a_t^2 & \ldots & a_t^{t-1} \end{pmatrix}$$

The determinant of the Vandermonde matrix is nonzero, provided that no two rows are identical, and it is found by the following formula :

$$|A_{\mathcal{S}}| = \prod_{i,j=1,i<j}^{t} (a_i - a_j) \tag{11}$$

Without loss of generality take $(a_1, a_2, \ldots, a_n) = (1, 2, \ldots, n)$. Obviously $\prod_{i,j=1,i<j}^{t} (a_i - a_j)$ divides $\prod_{i,j=1,i<j}^{n} (a_i - a_j)$. We also have

$$\prod_{i,j=1,i<j}^{n} (a_i - a_j) = 1^{\alpha_1} 2^{\alpha_2} \ldots (n-1)^{\alpha_{n-1}} \tag{12}$$

for some $\alpha_1, \alpha_2, \ldots, \alpha_{n-1}$. Hence by choosing e as a prime greater than n we can guarantee that the determinant of any $A_{\mathcal{S}}$ will be relatively prime to e.

VI. Conclusion

We presented an RSA threshold signature scheme based on Blakley secret sharing. To the best of our knowledge, this is the first threshold RSA signature scheme that uses Blakley SSS as the underlying secret sharing scheme. The scheme is as efficient as Shoup's practical threshold RSA signature and can be easily implemented. Moreover, this approach can be extended to other public key cryptosystems where the private key is used in the exponent.

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. Some recent research aspects of threshold cryptography. In *Proc. of ISW '97, 1st International Information Security Workshop*, volume 1196 of *LNCS*, pages 158–173. Springer-Verlag, 1997.

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

[5] 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.

[6] Y. Desmedt and Y. Frankel. Homomorphic zero-knowledge threshold schemes over any finite abelian group. *SIAM Journal on Discrete Mathematics*, 7(4):667–679, 1994.

[7] P. A. Fouque, G. Poupard, and J. Stern. Sharing decryption in the context of voting or lotteries. In *Proc. of FC 2000, 4th International Conference on Financial Cryptography*, volume 1962 of *LNCS*, pages 90–104. Springer-Verlag, 2001.

[8] H. F. Huang and C. C. Chang. A novel efficient (t,n) threshold proxy signature scheme. *Information Sciences*, 176(10):1338–1349, 2006.

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

[10] A. Lysyanskaya and C. Peikert. Adaptive security in the threshold setting: From cryptosystems to signature schemes. In *Proc. of ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 331–350. Springer-Verlag, 2001.

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

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

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