

*Function Sharing Schemes based on the  
Chinese Remainder Theorem*

Ali Aydın Selçuk

Bilkent University  
Ankara, Turkey

September 22, 2010  
Kingston University

# Outline

1. Threshold cryptography
2. Secret sharing schemes
3. Function sharing problem
4. A function sharing scheme based on the CRT
  - a. Basic solution
  - b. Provably secure version
5. Conclusions

# Overview

Threshold Cryptography:

## *Secret Sharing*

Three major schemes:

- Shamir
- Blakley
- Asmuth-Bloom

## *Function sharing*

- Traditionally based on Shamir; occasionally on Blakley.
- So far no solutions with Asmuth-Bloom.

This work: Achieving function sharing using the A-B SS scheme.

# *Secret Sharing*

How to share a sensitive secret  $d$  among  $n$  parties s.t. only a certain number  $t$  of them can together construct the secret?

## *Applications*

- Storage of sensitive cryptographic keys  
(e.g. root key in a PKI system)
- Command & control of nuclear weapons

## Secret Sharing

E.g. An  $(n, n)$  secret sharing scheme:

To share an  $\ell$ -bit secret key  $d$ ,

- generate random  $\ell$ -bit  $y_i$ ,  $i = 1, \dots, n - 1$ ,
- $y_n = d \oplus y_1 \oplus y_2 \oplus \dots \oplus y_{n-1}$
- give  $y_i$  to the  $i$ th user.

This scheme is *perfect*: A coalition smaller than  $t$  obtains no information.

**Q:** How to generalize to arbitrary  $(t, n)$ ?

## Shamir Secret Sharing Scheme

Fact: A degree- $(t - 1)$  polynomial is uniquely identified by  $t$  given points.

Shamir's  $(t, n)$ -threshold scheme to share  $d$ :

- Choose large prime  $p > d, n$
- Generate random polynomial in  $\mathbb{Z}_p[x]$

$$f(x) = a_{t-1}x^{t-1} + \dots + a_1x + a_0$$

where  $a_0 = d$ .

- Share of user  $i$  is  $y_i = f(x_i)$  for a random  $x_i$ .

## Shamir Secret Sharing Scheme

Secret construction by Lagrange interpolation:

$$f(x) = \sum_{i \in \mathcal{S}} y_i \prod_{\substack{j \in \mathcal{S} \\ j \neq i}} \frac{x - x_j}{x_i - x_j}$$

for  $|\mathcal{S}| \geq t$ .

Remark: Shamir SSS is perfect.

## Blakley Secret Sharing Scheme

Fact: In  $t$ -dimensional space, a system of  $t$  (non-degenerate) hyperplanes intersect at a single point.

Blakley's  $(t, n)$ -threshold scheme:

- Choose large prime  $p > d, n$
- Choose a random point  $x' = (x'_1, \dots, x'_t)$  in  $\mathbb{Z}_p^t$  s.t.  $x'_1 = d$ .
- Give the  $i$ th user a random hyperplane

$$a_{i1}x_1 + \dots + a_{it}x_t = y_i,$$

$1 \leq i \leq n$ , each passing through  $x'$ .

## *Blakley Secret Sharing Scheme*

Secret construction is by solving the system

$$A_S x = y_S.$$

Remark: Shamir's is a special case where

$$a_{ij} = \alpha_i^{j-1}$$

for some random  $\alpha_i$ .

I.e., where each  $A_S$  is Vandermonde.

## *Function Sharing Problem*

- Combiner platform may not be trusted.  
Users would rather not reveal their shares.
- Can we share the function rather than the private key, so that each user can compute his part without revealing his share?

## *Example Application: RSA Signature*

RSA signature on input  $w$ :

$$w^d \bmod N$$

for private key  $d$ , modulus  $N$ .

An  $(n, n)$  RSA signature sharing:

$$d = d_1 + \dots + d_n \bmod \phi(N)$$

where  $d_i, i \leq n - 1$ , are randomly generated.

## Sharing the RSA Signature

To compute signature  $w^d \bmod N$ , the  $i$ th user computes

$$w^{d_i} \bmod N.$$

The partial results are combined as

$$\begin{aligned} \prod_i w^{d_i} \bmod N &= w^{\sum_i d_i} \bmod N \\ &= w^d \bmod N \end{aligned}$$

**Q:** How to generalize to arbitrary  $(t, n)$ ?

## Function Sharing with Shamir

Note that

$$d = f(0) = \sum_{i \in \mathcal{S}} y_i \prod_{\substack{j \in \mathcal{S} \\ j \neq i}} \frac{0 - x_j}{x_i - x_j}$$

Hence,

$$w^d = \prod_{i \in \mathcal{S}} w^{y_i} \prod_j \frac{0 - x_j}{x_i - x_j}$$

where  $x_i$  are publicly known among members;  $y_i$  are secret.

Only if we could do Lagrange interpolation in  $\mathbb{Z}_{\phi(N)} \dots$

## *Function Sharing with Shamir*

Only if we could do Lagrange interpolation in  $\mathbb{Z}_{\phi(N)}$ :

- $(x_i - x_j)^{-1}$  may not exist in  $\mathbb{Z}_{\phi(N)}$ .
- Moreover,  $\phi(N)$  should not be known.

## *Solutions to FS with Shamir*

Ingenious solutions needed:

- *Desmedt & Frankel (1991)*
  - Dealer computes all inverses.
  - Distributes them mixed with shares.
  - Not so efficient.
- *De Santis, Desmedt, Frankel, Yung (1994)*
  - Arithmetic is taken to  $R = \mathbb{Z}(u)$  where  $u$  is a cyclotomic root of prime order.
  - $x_i \in R$  are chosen s.t. inverses exist for all  $(x_i - x_j)$ .
  - Inverses computed in  $\mathbb{Z}(u)$  are also inverses in  $\mathbb{Z}_{\phi(N)}(u)$ .
- *Shoup (2000)*
  - No need to take inverses nor to know  $\phi(N)$ .
  - Very simple and practical.

## Shoup's Th.RSA Scheme (Simplified)

- Take  $x_i = i$  (for  $i = 1, 2, \dots, n$ ).
- The  $i$ th user computes his partial signature as

$$s_i = w^{y_i n! \prod_j \frac{0-x_j}{x_i-x_j}} \bmod N$$

(i.e. with an extra  $n!$  )

- Combined as

$$s' = \prod_i s_i = w^{n!d} \bmod N.$$

- $n!$  can be conveniently removed to get

$$s = w^d \bmod N$$

for suitably chosen  $e$ . ( $\gcd(e, n!) = 1$ )

BREAK!

## *Chinese Remainder Theorem*

Given co-prime  $m_1, m_2, \dots, m_n$ , the system of equations

$$x \equiv x_1 \pmod{m_1}$$

$$x \equiv x_2 \pmod{m_2}$$

...

$$x \equiv x_n \pmod{m_n}$$

has a unique solution mod  $m = m_1 m_2 \cdots m_n$ .

E.g. Given

$$x \equiv 2 \pmod{3}$$

$$x \equiv 4 \pmod{5}$$

$$x \equiv 3 \pmod{4}$$

we have

$$x \equiv 59 \pmod{60}.$$

## Asmuth-Bloom Secret Sharing Scheme

- Choose integers  $m_0 < m_1 < \dots < m_n$  s.t.
  - $m_i$  are relatively prime
  - $m_0 > d$  is a prime
  - $m_i$  satisfy (for perfectness)

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

- Let  $M = \prod_{i=1}^t m_i$ . Compute

$$y = d + am_0$$

where  $a$  is some random integer s.t.  $0 \leq y < M$ .

- Share of the  $i^{\text{th}}$  user is

$$y_i = y \bmod m_i.$$

## *A-B SSS: Secret Construction*

- Given the system

$$y \equiv y_i \pmod{m_i}$$

for  $i \in \mathcal{S}$ , solve  $y$  in  $\mathbb{Z}_{M_{\mathcal{S}}}$  using the CRT.

(Note that  $M_{\mathcal{S}} \geq M$  for all valid  $\mathcal{S}$ .)

- Compute the secret as

$$d = y \bmod m_0.$$

## Function Sharing with Asmuth-Bloom

First, an additive solution for CRT:

1. Let  $M'_{S,i} = M_{S \setminus \{i\}}^{-1} \pmod{m_i}$ . User  $i$  computes

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

2.  $y$  is computed as

$$y = \sum_{i \in S} u_i \pmod{M_S}.$$

3. The secret  $d$  is computed as

$$d = y \pmod{m_0}.$$

Note that  $m_0$  is not needed until Step 3.

Also note that  $m_0$  need not be prime.

## Sharing RSA with Asmuth-Bloom

Secret construction is additive

$$\sum_{i \in S} y_i M'_{S,i} M_{S \setminus \{i\}} \pmod{M_S}$$

hence may be suitable to share RSA:

$$w^d \pmod{N} = \prod_{i \in S} w^{y_i \dots} \pmod{N}$$

Challenge: But how to include  $(\pmod{M_S})$  in the exponent?

(Shamir and Blakley works with the same modulus  $p$  for all coalitions, which was made related to  $\phi(N)$ . Here, each coalition has a different  $M_S$ . How to relate  $M_S$  to  $\phi(N)$ ?)

## A FSS for RSA Signatures

1. RSA setup with  $p = 2p' + 1$ ,  $q = 2q' + 1$ .  
 $N = pq$ ;  $ed \equiv 1 \pmod{\phi(N)}$ .  
Use A-B to share  $d$  with  $m_0 = \phi(N) = 4p'q'$ .
2. To sign  $w$ , user  $i \in \mathcal{S}$  computes

$$u_i = y_i M'_{\mathcal{S},i} M_{\mathcal{S} \setminus \{i\}} \pmod{M_{\mathcal{S}}},$$
$$s_i = w^{u_i} \pmod{N}.$$

3. The *incomplete signature*  $\bar{s}$  is

$$\bar{s} = \prod_{i \in \mathcal{S}} s_i \pmod{N}.$$

## A FSS for RSA Signatures

Now  $\bar{s} = \prod_{i \in S} s_i$  has some extra  $M_S$  to be removed. I.e.

$$\bar{s} = w^{d+jS} \text{ for some } 0 \leq j \leq t-1.$$

Hence the correction step:

4. Let  $\lambda = w^{-M_S} \pmod N$  be the *corrector*. Try

$$(\bar{s}\lambda^j)^e = \bar{s}^e(\lambda^e)^j \stackrel{?}{\equiv} w \pmod N \quad (1)$$

for  $0 \leq j < t$ . Then the signature  $s$  is

$$s = \bar{s}\lambda^\delta \pmod N$$

where  $\delta$  is the  $j$  value satisfying (1).

## A FSS for RSA Signatures

*Remarks:*

1. In the exponent,

$$\sum_{i \in S} u_i = y + \delta M_S$$

for some  $1 \leq \delta \leq t - 1$ .

We add in the “corrector”  $\lambda = w^{-M_S}$  until we obtain  $w^y$ .

2. When we obtain  $y$  in the exponent, we have

$$w^y \equiv w^d \pmod{N}$$

since  $y = d + am_0$  where  $m_0 = \phi(N)$ .

3. Given that  $n \ll N$  there is a negligible chance that there is a second  $j$  that satisfies the equality.

## *Comparison to Earlier Works*

- *Performance:*
  - User:  $t + 1$  multiplications, one inversion, one exponentiation
  - Combining:  $t - 1$  multiplications
  - Correction: one exponentiation, up to  $t - 1$  multiplications
- Better than Desmedt & Frankel (1991) and De Santis et al. (1994).

## *Further Work*

- Provably secure version (Reduction: If this scheme can be broken, then the RSA signature scheme can be broken.)
- Verifiability: If a user or dealer cheats, it can be detected.
- Proactivity: Shares can be updated without affecting the secret key.
- Robustness The function can be computed correctly even if some users cheat (up to a certain number).
- Protocols for other PKC systems: ElGamal, Paillier, etc.

## *Concluding Remarks*

- Up until now, function sharing has traditionally been based on Shamir or Blakley. We showed how it can be done with Asmuth-Bloom.
- The schemes are comparable to the earlier works performance-wise.