Generic Homomorphic Undeniable Signature Scheme: Optimizations

Yvonne Anne Oswald

Semester Project February 2005

Responsible
Prof. Serge Vaudenay

Supervisor
Jean Monnerat

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

LASEC

Outline

- Introduction generic homomorphic undeniable signatures
- Homomorphisms: quartic residue symbol, Jacobi symbol, discrete logarithm, RSA
- Comparison signature generation
- Conclusion

Undeniable Signatures

- Verification of validity of signature only by interaction with Signer
- Complete signature scheme consists of
  - Key generation algorithm
  - Signature algorithm
  - Confirmation protocol
  - Denial Protocol

GHI Problem

Definition: G, H Abelian groups, $S := \{(x_1, y_1), \ldots, (x_s, y_s)\} \subseteq G \times H$. We say that S interpolates in a group homomorphism if there exists a homomorphism $\phi : G \rightarrow H$ such that $\phi(x_i) = y_i$ for $i = 1, \ldots, s$.

GHI Problem $\big($Group Homomorphism Interpolation Problem$\big)$
Parameters: two Abelian Groups G and H, a set of s points $S \subseteq G \times H$.
Input: $x \in G$
Problem: find $y \in H$ such that $\big(x, y\big)$ interpolates with S in a group homomorphism.

The GHI problem is the generalization of many cryptographic problem: discrete logarithm, Diffie-Hellmann, RSA.

Generic homomorphic undeniable signatures

Key generation:

- Select G, H Abelian groups and hom. $\phi : G \to H$ (private key)
- Compute public key $K := \{(x_{key1}, \phi(x_{key1}), \ldots, (x_{keyk}, \phi(x_{keyk}))\}$, x_i generated from seed ρ using det. pseudorandom generator

Signature and protocols:

- Generate $(x_1, \ldots, x_s)$ from message m using det. pseudorandom generator
- Compute signature $S := \{(x_{sig1}, \phi(x_{sig1}), \ldots, (x_{sigs}, \phi(x_{sigs}))\}$
- Confirmation/denial protocol: proving/disproving that K interpolates with S in a homomorphism

Generic homomorphic undeniable signatures

- Security of generic homomorphic undeniable signatures depends on hardness of GHI problem
- Various homomorphisms suitable (hard characters, discrete logarithm, RSA exponentiation)
- Summer 2004: Demonstrator for MOVA signature scheme using quartic residue symbol as homomorphism

Description of project

- Optimize existing implementation of quartic residue symbol
- Implement 3 additional homomorphisms (Jacobi symbol, discrete logarithm, RSA exponentiation) and compare them to each other
- Programming language C, GNU Multiple Precision Arithmetic Library GMP

Quartic residue symbol $\chi_\beta(\alpha)$

Definition Let $\alpha, \beta \in \mathbb{Z}[i]$ be such that $(1+i) \nmid \beta$ and $\gcd(\beta, \alpha) = 1$. The quartic residue symbol is defined as $\chi_\beta : \mathbb{Z}[i] \to \{0, \pm 1, \pm i\}$

$$\chi_\beta(\alpha) = \begin{cases} (\alpha^{\frac{N(\beta)-1}{4}}) \bmod \beta & \text{if } \beta \text{ prime} \\ \prod_i \ \chi_{\beta_i}(\alpha) & \text{if } \beta = \prod_i \beta_i, \ \beta_i \text{ prime} \end{cases}$$

Properties

- Modularity: $\chi_\beta(\alpha) = \chi_\beta(\alpha \bmod \beta)$
- Multiplicativity: $\chi_\beta(\alpha\alpha') = \chi_\beta(\alpha)\chi_\beta(\alpha')$
- Reciprocity Law: if $\alpha, \ \beta$ primary:
$$\chi_\beta(\alpha) = \chi_\alpha(\beta) \cdot (-1)^{\frac{N(\alpha)-1}{4} \cdot \frac{N(\beta)-1}{4}}$$

Quartic residue symbol $\chi_\beta(\alpha)$

Basic algorithm

$t \leftarrow 0$

WHILE $(N(\alpha) > 1)$ DO

$\alpha \leftarrow \alpha \bmod \beta$ (Modularity)

find α' primary such that $i^j \cdot (1+i)^k \cdot \alpha'$

set $\alpha \leftarrow \alpha'$ and adjust t (Multiplicativity)

swap α and β and adjust t (Reciprocity)

IF $(N(\alpha) = 1)$ return i^t

Quartic residue symbol χ

Damgård's algorithm

$t \leftarrow 0$

WHILE $(\alpha \neq \beta)$ DO

$\alpha \leftarrow \alpha - \beta$

find α' primary such that $i^j \cdot (1+i)^k \cdot \alpha'$

$\alpha \leftarrow \alpha'$ and adjust t (Multiplicativity)

IF $(N(\alpha) > N(\beta))$

swap α and β and adjust t (Reciprocity)

IF $(\alpha = 1)$ return i^t

Quartic residue symbol χ

Implementation

Basic functions in $\mathbb{Z}[i]$

- Multiplication: $\alpha \cdot \beta$
- Modulo: $\alpha \bmod \beta$
- Norm: $N(\alpha)$
- Division by $(1+i)^r$
- Primarization: transforms α into its primary associate if possible

Quartic residue symbol χ

Optimization

- Scrutinize every line of code
- Implement faster algorithms
- Remove unnecessary function calls
- Use faster, more sophisticated GMP functions
- Reduce the number of GMP variables
- Examine different implementation variants
- Apply general optimization techniques

Number of lines of code: not optimized 1162, optimized 603

Division by $(1+i)^r$

$$\frac{\alpha}{(1+i)} = \frac{Re(\alpha)+Im(\alpha)}{2} + \frac{Im(\alpha)-Re(\alpha)}{2}i$$

$$\frac{\alpha}{(1+i)^r} = \frac{i^{3k}\left(\frac{Re(\alpha)}{2^k} + \frac{Im(\alpha)}{2^k}i\right)}{(1+i)^b} \quad , \quad r = 2k+b$$

Not optimized:

- Compute $(1+i)^r$
- Divide α by result

Optimized:

- Rightshift $Re(\alpha)$ and $Im(\alpha)$ by $k+b$ considering $3k$
- If $b=1$ perform necessary addition and subtraction

Quartic residue symbol χ

Tests

- Time measurements under Windows and Linux
- Windows platform Intel(R)4 1.4 GHz Desktop Computer,
  256 MB RAM, Windows XP
- Average values produced by test series of 1000 tests
- 512 bit random numbers generated with a GMP function
  Gaussian integers: real and imaginary part 512 bit random numbers

Quartic residue symbol χ

Results subfunctions

Running time (in ms)	not optimized	optimized	GMP (in $\mathbb{Z}$)
Multiplication in $\mathbb{Z}[i]$	0.078	0.049	0.010
Modulo in $\mathbb{Z}[i]$	0.141	0.104	0.001
Division by $(1+i)^r$	0.061	0.015	0.001
Primarization	0.071	0.006	

Quartic residue symbol χ

Results quartic residue symbol

Running time and iterations	time in ms	iterations
Basic algorithm not optimized	62.79	249.27
Basic algorithm optimized	31.57	249.27
Damgård's algorithm	24.22	512.84

Quartic residue symbol χ

MOVA signature scheme

- $n = pq$ where $p,\ q$ rational primes, $p \equiv q \equiv 1 \pmod 4$
- Find π, σ such that $p = \pi\bar{\pi}$ and $q = \sigma\bar{\sigma}$ with algorithms by Tonelli and Cornacchia
- Select $G := \mathbb{Z}[i]/\beta\mathbb{Z}[i],\ G \cong \mathbb{Z}_n^*$ with $\beta = \pi\sigma$, $G \cong \mathbb{Z}_p^*$ with $\beta = \pi$

Quartic residue symbol χ

Results MOVA Setup

Running time and iterations	time in ms	iterations
$\beta = \pi\sigma$		
Basic algorithm optimized	32.12	248.81
Damgård's algorithm	50.63	766.12
$\beta = \pi$		
Basic algorithm optimized	14.31	2123.87
Damgård's algorithm	38.59	640.71

Quartic residue symbol χ

Mixed Algorithm

$\alpha \leftarrow \alpha \bmod \beta$

return Damgård(α, β)

Quartic residue symbol χ

Results including mixed algorithm

MOVA setup running time and iterations	time in ms	iterations
$\beta = \pi\sigma$		
Basic algorithm optimized	32.12	248.81
Damgård's algorithm	50.63	766.12
Mixed algorithm	24.65	511.92
$\beta = \pi$		
Basic algorithm optimized	14.31	2123.87
Damgård's algorithm	38.59	640.71
Mixed algorithm	9.03	255.95

Jacobi Symbol $\left(\frac{a}{m}\right)$

- Jacobi symbol equivalent of quartic residue symbol in $\mathbb{Z}$
- Implement basic algorithm (modulo for reduction)
- Compare with GMP function `mpz_jacobi` (binary algorithm)
- MOVA setup: $n = pq,\ p, q$ prime, $a \in \mathbb{Z}_n^*, m = p$

Results Jacobi symbol

Running time and iterations	time in ms	iterations
Basic algorithm	1.261	187.71
Binary algorithm (GMP)	0.116	

Homomorphism based on discrete logarithm

- $n = pq$ with $p = rd + 1$, q, d prime, $\gcd(q-1, d) = 1,\ \gcd(r, d) = 1$
- g generating a subgroup of $\mathbb{Z}_p^*$ of order d

Construct a homomorphism suitable for the generic homomorphic signature scheme by computing a discrete logarithm in a small subgroup of $\mathbb{Z}_n^*$ like this:

$$\phi : \mathbb{Z}_n^* \to \mathbb{Z}_d \quad \phi(x) = \log_g(x^r \bmod p)$$

GMP provides a function for exponentiation modulo an integer
$\longrightarrow$ only discrete logarithm to implement

Homomorphism based on discrete logarithm

Precomputed table

Construct table containing discrete logarithm for all elements

Size of table: 2^{20} 20 bit integers, key 512 bits long

- Hash table: array of integers (32 bits)
- Key management: key not saved, index into array
- Collision management: 24 LSB of key, key $>>$ 24, ...
- Insertion/Retrieval methods: collision check, correctness check
- Preprocessing time: $O(d)$, running time: $O(1)$

Homomorphism based on discrete logarithm

Baby step giant step algorithm (BSGS)

- Store $O(\sqrt{d})$ elements in hashtable (baby steps)
- Loop from 0 to $O(\sqrt{d})$ to find logarithm (giant steps)
- Use hash table from previous algorithm, less collisions
- Preprocessing time: $O(\sqrt{d})$, running time: $O(\sqrt{d})$

Homomorphism based on discrete logarithm

Pollard's rho algorithm

- Group G partitioned into three sets S_0, S_1, S_2 of roughly equal size
- Define sequence of group elements x_i and integers a_i, b_i satisfying $x_i = g^{a_i} y^{b_i}$ $\forall i \geq 0$ by $x_0 = 1,\ a_0 = 0,\ b_0 = 0$ and

$$x_{i+1} = \begin{cases} y x_i \bmod p & \text{if } x_i \in S_0 \\ x_i^2 \bmod p & \text{if } x_i \in S_1 \\ g x_i \bmod p & \text{if } x_i \in S_2 \end{cases}$$

- Compute iteratively x_i and x_{2i} until $x_i = x_{2i}$
- Derive logarithm from representation $x_i = g^{a_i} y^{b_i}$
- Running time: $O(\sqrt{d})$

Homomorphism based on discrete logarithm

Results

Table construction	time in s	collisions
Precomputed table	16.616	14 274
Baby step giant step	6.023	0

Running time and iterations	time in ms	iterations
GMP `mpz_powm`	9.44	
Precomputed table	9.66	1.04
Baby step giant step	19.47	388.36
Pollard's rho	74.92	1037.49

RSA

- Select $n = pq,\ p, q$ prime, $\phi(n) = (p-1)(q-1)$
- Choose e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$
- Compute $d = e^{-1} \bmod \phi(n)$

Homomorphism ϕ suitable for generic homomorphic signature scheme:

$$\phi : \mathbb{Z}_n^* \to \mathbb{Z}_n^* \quad \phi(x) = (x^d \bmod n)$$

Results

Running time	time in ms
RSA exponentiation	33.87

Signature Generation

Security parameters

- Quartic residue symbol n 1024 bits, $s = 20$
- Jacobi symbol n 1024 bits, $s = 20$
- Hom. based on discrete logarithm d 20 bits, n 1024 bits, $s = 1$
- RSA n 1024 bits, $s = 1$

Signature generation

Results

Signature generation time	time in ms
Quartic residue symbol $(\beta = \pi\sigma)$	493.01
Quartic residue symbol $(\beta = \pi)$	180.64
Jacobi symbol (basic algorithm)	25.22
Jacobi Symbol (GMP)	2.32
Discrete logarithm (Precomputed Table)	9.66
Discrete logarithm (BSGS)	19.47
Discrete logarithm (Pollard's rho)	74.93
RSA	33.87

Conclusion

- Reduced computation time for quartic residue symbol by more than half
- $\mathbb{Z}[i]$ computations prevent quartic residue symbol from being competitive with Jacobi symbol
- Compared implementations of additional homomorphisms by storage requirement and speed