Privacy Homomorphisms

Approaches, Implementation and Applications

Zeming Wang

Supervisor:
Prof. Zhenfu Cao

Department of Computer Science and Engineering
Shanghai Jiaotong University

June, 2013

Outline

Outline

Privacy Homomorphisms

A privacy homomorphism, or homomorphic encryption, is an encryption transformation which allows the encrypted data to be operated on without knowledge of the decryption function.

Formal Definition

A *privacy homomorphism* is a homomorphism ϕ from an algebraic system U consisting of a set S, some operations $f_1, f_2, ...$, some predicates $p_1, p_2, ...$, and some distinguished contants $s_1, s_2, ...$, to an algebraic system C consisting of a set S', some operations $f_1', f_2', ...$, some predicates $p_1', p_2', ...$, and some distinguished contants $s_1', s_2', ...$, such that:

1. $(\forall i)(a, b, c, ...) \in S', f_i'(a, b, ...) = c \Rightarrow f_i(\phi(a), \phi(b), ...) = \phi(c)$
2. $(\forall i)(a, b, c, ...) \in S', p_i'(a, b, ...) \equiv p_i(\phi(a), \phi(b), ...)$
3. $(\forall i)\phi(s_i') = s_i$.

Outline

Types of Homomorphic Cryptosystems

- Homomorphic public-key cryptosystems
- Algebraic privacy homomorphisms
- Fully homomorphic encryptions

Homomorphic Public-Key Cryptosystems

RSA satisfies homomorphic multiplication.
If

$$c_1 = m_1^e \mod n$$

$$c_2 = m_2^e \mod n$$

Then

$$c_1 c_2 \mod n = m_1^e m_2^e \mod n = (m_1 m_2)^e \mod n$$

Homomorphic Public-Key Cryptosystems (Cont'd)

Cryptosystem	Security Assumption	Homomorphic Operations	Message Expansion
RSA	RSA Problem	$\boxtimes$	1
Goldwasser-Micali	Quadratic Residuosity Problem	XOR	n
ElGamal	CDH / DDH	$\boxtimes$	2
Benaloh	Weak r'th Root Problem	$\boxplus, \boxminus, \boxtimes_c$	$\frac{n}{r}$
Naccache-Stern	Factoring / DLP / Weak r'th Residue Problem	$\boxplus, \boxminus, \boxtimes_c$	≥ 4
Sander-Young-Yung	Quadratic Residuosity Problem	AND	kn
Okamoto-Uchiyama	Factoring / p-subgroup Problem	$\boxplus, \boxminus, \boxtimes_c$	3
Modified Okamoto-Uchiyama	Factoring / p-subgroup Problem	$\boxplus, \boxminus, \boxtimes_c$	3
Improved Okamoto-Uchiyama	Factoring / p-subgroup Problem	$\boxplus, \boxminus, \boxtimes_c$	3
Paillier	Class[n] / D-Class[n]	$\boxplus, \boxminus, \boxtimes_c$	2

Homomorphic Public-Key Cryptosystems (Cont'd)

Fast Decryption Paillier	PDL$[n]$ / D-PDL$[n]$	$\boxplus, \boxminus, \boxtimes_c$	2
Small Exponent Paillier	Small e'th Root Problem	$\boxplus, \boxminus, \boxtimes_c$	2
Modified Paillier	Class$[n]$ / D-Class$[n]$ on restricted generators	$\boxplus, \boxminus, \boxtimes_c$	2
Schmidt-Samoa-Takagi	Factoring Problem	$\boxplus, \boxminus, \boxtimes_c$	3
Elliptic Curve Paillier	Subgroup Decision Problem	$\boxplus, \boxminus, \boxtimes_c$	2
Damgård-Jurik	Class$[n^s]$ / D-Class$[n^s]$	$\boxplus, \boxminus, \boxtimes_c$	$\frac{s+1}{s}$
Length Flexible Damgård-Jurik	Class$[n^s]$ / D-Class$[n^s]$	$\boxplus, \boxminus, \boxtimes_c$	$\frac{s+1}{s}$
Modified Length Damgård-Jurik	Class$[n^s]$ / D-Class$[n^s]$	$\boxplus, \boxminus, \boxtimes_c$	$\frac{s+1}{s}$
Boneh-Goh-Nissim	Subgroup Decision Problem	$\boxplus, \boxminus, \boxtimes_c$ $\boxtimes$ (once)	$\frac{n}{r}$

Algebraic Privacy Homomorphisms

Initialization

Choose two secret large primes p, q, and let $n = pq$.

Encryption

Given cleartext $x \in \mathbf{Z}_n$, compute $(x \bmod p, x \bmod q)$.

Decryption

Given ciphertext (y_1, y_2), compute $x = CRT(y_1, y_2)$ with known p, q.

Fully Homomorphic Encryptions

Somewhat homomorphic scheme

- Define addition and multiplication on the ciphertext;
- Evaluate circuits of additions and multiplications up to a certain depth;
- Somewhat homomorphic scheme, because of noise.

Fully homomorphic scheme

- Evaluate its own decryption circuit, i.e. bootstrappable;
- Noise reduced and achieve fully homomorphic.

Types of Homomorphic Cryptosystems

Among the above three types, we think the algebraic privacy homomorphism is the most practical solution for homomorphic computing. So we focus on *symmetric-key algebraic privacy homomorphisms* in this paper.

Outline

Doming-Ferrer's First Approach

Initialization

Choose two large secret primes p, q, compute $n = pq$; choose a positive integer d; choose two random integers $r_p \in \mathbf{Z}_p^*, r_q \in \mathbf{Z}_q^*$.

Encryption

Giver cleartext $x \in \mathbf{Z}_n$, randomly split it into $x_1, x_2, \ldots, x_d$ such that

$$x = \sum_{i=1}^{d} x_i \mod n, x_i \in \mathbf{Z}_n$$

Compute

$$X = ([x_1 r_p \bmod p, x_1 r_q \bmod q], \ldots, [x_d r_p^d \bmod p, x_d r_q^d \bmod q])$$

as ciphertext.

Doming-Ferrer's First Approach (Cont'd)

Decryption

Given ciphertext $X = [x_j r_p^j \bmod p, x_j r_q^j \bmod q]$, multiply it by $[r_p^{-j} \bmod p, r_q^{-j} \bmod q]$, and get

$$([x_1 \bmod p, x_1 \bmod q], ..., [x_d \bmod p, x_d \bmod q])$$

Sum them up

$$[x \bmod p, x \bmod q] = \sum_{i=1}^{d} [x_i \bmod p, x_i \bmod q]$$

Then compute x using CRT.

Outline

New Approach

Initialization

Choose two large secret primes p, p', with $p < p'$, compute $n = pp'$. p, p' are secret, while n is public.

Encryption

Given cleartext $x \in \mathbf{Z}_p$, compute ciphertext

$$y = E(x) = x^p \mod n.$$

Decryption

Given ciphertext $c \in \mathbf{Z}_n$, compute cleartext

$$x = D(x) = y \bmod p.$$

Proof of Correctness

Theorem

For all $x \in \mathbf{Z}_p$, it holds that $D(E(x)) = x$.

Proof

By definition,

$$y \equiv x^p \mod n$$

Since $p \mid n$, we have

$$y \equiv x^p \mod p$$

By Fermat's Little Theorem,

$$y \equiv x^p \equiv x \mod p$$

Since $x < p$, it holds that

$$x = y \bmod p.$$

Homomorphic Properties

Multiplication

It is obvious the scheme provides homomorphic multiplication.

Addition

Suppose $c = c_1 + c_2 \equiv m_1^p + m_2^p \pmod{n}$
Since $n = pq$, $p \mid n$, so we have

$$c \equiv m_1^p + m_2^p \pmod{p}$$

And the following equation holds,

$$c \equiv m_1^p + m_2^p \equiv (m_1 + m_2)^p \pmod{p}$$

By Fermat's little theorem,

$$(m_1 + m_2)^p \equiv m_1 + m_2 \pmod{p}$$

Therefore it finally gives $m_1 + m_2 = c \mod p$.

Homomorphic Properties (Cont'd)

Substraction

Similar to addition.

Multiplicative Inverse

Since p is a prime, for any $x \in \mathbf{Z}_p$, the multiplicative inverse of x must exist. And $p' > p$, so $gcd(x, p') = 1$, and it follows that $gcd(x, n) = 1$. Therefore $gcd(y, n) = 1$, which indicates that the multiplicative inverse of y exists on $\mathbf{Z}_n$. So we have

$$y^{-1} \equiv (x^p)^{-1} \equiv \left(x^{-1}\right)^p \mod n$$

Since $p|n$, it follows that

$$y^{-1} \equiv \left(x^{-1}\right)^p \equiv x^{-1} \mod p$$

Outline

Numeric Example

- Suppose the aim is to compute $(2+3+4) \times 2/3$.
- In the classified level, let $p = 17, q = 19$, and $n = pq = 323$, encrypt all the data:

$$E(2) = 2^{17} \bmod 323 = 257$$
$$E(3) = 3^{17} \bmod 323 = 241$$
$$E(4) = 4^{17} \bmod 323 = 157$$

  Send $E(2), E(3), E(4)$ and $n = 323$ to unclassified level.
- The unclassified level compute on encrypted data:

$$(257 + 241 + 157) \times 257 \times 241^{-1} \equiv 23 \mod 323$$

  and retuen 23.
- The classified use secret key $p = 17$ to decrypt and get the result $23 \bmod 17 = 6$.

Outline

Known-Cleartext Attacks

Suppose the cryptanalyst knows k pairs, namely, $(x_i, y_i), i = 1, 2, ..., k$. Take the greatest common divisor of the difference between the ciphertext and cleartext, and let it be

$$\hat{p} = gcd\{y_i - x_i : i = 1, 2, ..., k\}$$

Since $y_i \equiv x_i \mod p$, we have $p|(y_i - x_i)$. So p must divide the greatest common divisor, i.e. $p|\hat{p}$. Even for a small k, there is a high probability of $\hat{p} = p$. If this is not the case, every new pair lets the attacker come closer to the secret prime.

Variation

Given cleartext $x \in \mathbf{Z}_m$. Secretly and randomly split x into k integers $x_0, x_2, \ldots, x_{k-1}$, such that $x = \sum_{i=0}^{k-1} x_i$. Randomly choose a secret r outside $\mathbf{Z}_m$. Define

$$\theta(\mathbf{x}) = \theta(x_0, x_2, \ldots, x_{k-1}) = \sum_{i=0}^{k} x_i r^i$$

Notice that this mapping preserves addition, substraction and multiplication, because

$$x \pm y = \sum_{i=0}^{k-1} (x_i \pm y_i) \leftrightarrow \theta(\mathbf{x}) \pm \theta(\mathbf{y})$$

$$xy = \sum_{l=0}^{2(k-1)} \sum_{i+j=l} x_i y_j \leftrightarrow \theta(\mathbf{x})\theta(\mathbf{y})$$

Variation (Cont'd)

Encryption

Given cleartext $x \in \mathbf{Z}_m$. Do the following to encrypt:

1. Randomly split x into k integers $x_0, x_2, \ldots, x_{k-1}$, s.t. $x = \sum_{i=0}^{k-1} x_i$;
2. Randomly choose r outside $\mathbf{Z}_m$, and compute $y = \theta(x_0, x_2, \ldots, x_{k-1})$;
3. Choose a secret prime p, such that $L(p) > \alpha L(r^k)$, and compute $z = E_p(y)$;

Then z is the ciphertext.

Decryption

Given ciphertext z. Do the following to decrypt:

1. Compute $y = D(z)$;
2. Compute $\mathbf{x} = \theta^{-1}(y)$;
3. Compute $x = \sum_{i=0}^{k-1} x_i$.

And x is the decrypted cleartext.

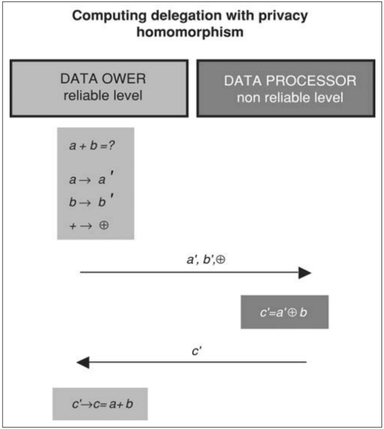
Security Improvements

Due to the use of θ, the cleartext is no longer congruent to the ciphertext. So the greatest common divisor cryptanalysis method does not work here.

Outline

Computing Delegation

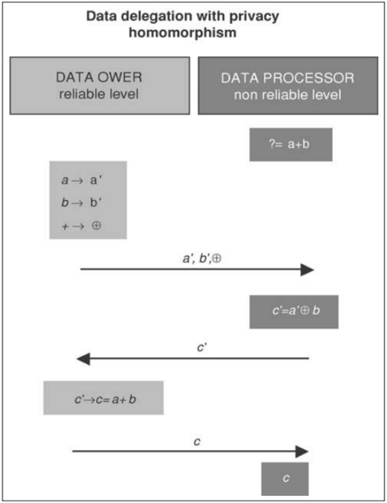
In computing delegation, the data processor only deals with encrypted data.



Outline

Computing Delegation

In data delegation, the data processor requires the decrypted result.



Conclusion

- It seems to be a fact that, the more homomorphic operations a privacy homomorphism supports, the lower its security level will be; and if the security is strengthened, the efficiency will be sacrificed.

Conclusion

- It seems to be a fact that, the more homomorphic operations a privacy homomorphism supports, the lower its security level will be; and if the security is strengthened, the efficiency will be sacrificed.
- There is a dilemma: to encrypt something means to make it diffused and to remove order; while a homomorphism means to preserve some sort of order in the ciphertext.

Conclusion

- It seems to be a fact that, the more homomorphic operations a privacy homomorphism supports, the lower its security level will be; and if the security is strengthened, the efficiency will be sacrificed.
- There is a dilemma: to encrypt something means to make it diffused and to remove order; while a homomorphism means to preserve some sort of order in the ciphertext.
- In future work, the creation of a good algebraic privacy homomorphism, i.e. a homomorphic cryptosystem which is both secure and practical, is still very challenging and prominent!

The End

Thank You!