SHE AND FHE

Hammad Mushtaq

ENEE759L

March 10, 2014

Outline

- Introduction
  - Needs
  - Analogy
- Somewhat Homomorphic Encryption (SHE)
  - RSA , EL GAMAL (MULT)
  - Pallier (XOR and ADD)
- Fully Homomorphic Encryption (FHE)
  - Protocol
  - Bootstrapping

Introduction

- Needs for Fully Homomorphic Encryption (FHE)
  - Secure Cloud Computing
  - Ability to compute encrypted data over public servers, while the ciphertext is decrypted to retrieve the correct computed data
  - Encrypted search queries in email, search engines, etc.

Introduction Continued…

- Basic Encryption Involves 3 Steps
  - KeyGen
  - Encrypt
  - Decrypt

- Fully Homomorphic Encryption
  - Keygen
  - Encrypt
  - Evaluate
  - Decrypt

Analogy

- You want to compute the total cash available across all your assets in one place (checking, investments, real estate properties, etc.)
- You do not want the server to know this valuable information, but you need the server to aggregate all this information and compute the sum

PROBLEM: HOW DO WE KEEP THE COMPUTATION ENCRYPTED WITHOUT GIVING UP IMPORTANT CHARACTERISTICS TO THE SERVER?

Somewhat Homomorphic Encryption (SHE) for Multiplying

- RSA
  - Using this method, you can multiply two encrypted numbers over the cloud without revealing any information about the plaintext
  - Although it is possible to add two encrypted numbers, it reveals enough information for the adversary to exploit it
  - El Gamal can also provide the same functionality as mentioned

The Homomorphism: Suppose x_1 and x_2 are plaintexts. Then,

$$e_K(x_1)e_K(x_2) = x_1^b\, x_2^b \bmod n = (x_1 x_2)^b \bmod n = e_K(x_1 x_2)$$

RSA Example

1. Choose two distinct prime numbers, such as
   $$p = 61 \text{ and } q = 53$$
2. Compute $n = pq$ giving
   $$n = 61 \times 53 = 3233$$
3. Compute the totient of the product as $\phi(n) = (p-1)(q-1)$ giving
   $$\varphi(3233) = (61-1)(53-1) = 3120$$
4. Choose any number $1 < e < 3120$ that is coprime to 3120. Choosing a prime number for e leaves us only to check that e is not a divisor of 3120.
   Let $e = 17$
5. Compute d, the modular multiplicative inverse of e (mod $\phi(n)$) yielding
   $$d = 2753$$

The **public key** is ($n = 3233$, $e = 17$). For a padded plaintext message m, the encryption function is

$$c(m) = m^{17} \mod 3233$$

The **private key** is ($n = 3233$, $d = 2753$). For an encrypted ciphertext c, the decryption function is

$$m(c) = c^{2753} \mod 3233$$

For instance, in order to encrypt $m = 65$, we calculate

$$c = 65^{17} \mod 3233 = 2790$$

To decrypt $c = 2790$, we calculate

$$m = 2790^{2753} \mod 3233 = 65$$

Somewhat Homomorphic Encryption (SHE) for Addition

- Pallier
  - Allows for ADD and XOR functions over the cloud
  - Can do so without compromising important information about the plaintext
  - Can also do multiplication, but the same problem arises with RSA using addition

Paillier Protocol

Key generation [edit]

1. Choose two large prime numbers p and q randomly and independently of each other such that $\gcd(pq, (p-1)(q-1)) = 1$. This property is assured if both primes are of equivalent length, i.e., $p, q \in 1||\{0,1\}^{s-1}$ for security parameter s.[1]
2. Compute $n = pq$ and $\lambda = \operatorname{lcm}(p-1, q-1)$.
3. Select random integer g where $g \in \mathbb{Z}^*_{n^2}$
4. Ensure n divides the order of g by checking the existence of the following modular multiplicative inverse:

$$\mu = (L(g^\lambda \mod n^2))^{-1} \mod n,$$

where function L is defined as $L(u) = \frac{u-1}{n}$.

Note that the notation $\frac{a}{b}$ does not denote the modular multiplication of a times the modular multiplicative inverse of b but rather the quotient of a divided by b, i.e., the largest integer value $v \geq 0$ to satisfy the relation $a \geq vb$.

- The public (encryption) key is (n, g).
- The private (decryption) key is (λ, μ).

If using p,q of equivalent length, a simpler variant of the above key generation steps would be to set $g = n + 1, \lambda = \varphi(n)$, and $\mu = \varphi(n)^{-1} \mod n$, where $\varphi(n) = (p-1)(q-1)$.[1]

Paillier Protocol (cont.)

Encryption [edit]

1. Let m be a message to be encrypted where $m \in \mathbb{Z}_n$
2. Select random r where $r \in \mathbb{Z}_n^*$
3. Compute ciphertext as: $c = g^m \cdot r^n \mod n^2$

Decryption [edit]

1. Let c be the ciphertext to decrypt, where $c \in \mathbb{Z}_{n^2}^*$
2. Compute the plaintext message as: $m = L(c^\lambda \mod n^2) \cdot \mu \mod n$

- **Homomorphic addition of plaintexts**

The product of two ciphertexts will decrypt to the sum of their corresponding plaintexts,

$$D(E(m_1, r_1) \cdot E(m_2, r_2) \mod n^2) = m_1 + m_2 \mod n.$$

The product of a ciphertext with a plaintext raising g will decrypt to the sum of the corresponding plaintexts,

$$D(E(m_1, r_1) \cdot g^{m_2} \mod n^2) = m_1 + m_2 \mod n.$$

Fully Homomorphic Encryption

Craig Gentry - Theorized a scheme that solved this open problem. His approach concentrated more on possibility rather than practicality

DGHV - Gentry, Halevi + 2 Others -> Developed on Gentry's initial findings to develop a conceptually simpler scheme

Gentry' Scheme: Because of its complexity and non-security focus, it was hard to suggest guidelines for the parameters that would be used

DGHV Scheme: Public Key Size: 2^60 bits =

1.1529215e+18

Continued...

For Gentry's scheme: improvement by Smart and Vercauteren (PKC 2010); implementation by Gentry and Halevi (Eurocrypt 2011). PK size: 2:3 GB. Ciphertext refresh: 30 minutes.

Gentry’s Scheme was Improved Upon by Smart and Vercauteren (PKC 2010) and was implemented by Gentry and Halevi (Eurocrypt 2011)

Public Key Size: 2.3 Gigabytes -> Ciphertext Refresh 30 Minutes

DGHV: Public Key Size 800 MB -> Ciphertext Refresh: 15 Minutes

DGHV Construction

- This construction is based on the hardness of the approximate greatest common divisors problem
- Given polynomial $x_i = pq_i + r_i$ such numbers
  - p is a random odd number chosen from $[0, 2^a]$
  - q_i is chosen from a Distribution $D = D_p$ which can depend on p
  - r_i is some random noise from $[-2^b, 2^b]$
    - a, b and D are parameters of the problem
  - if there is random noise r_i it is hard to find p
  - Keep p as the secret key,
- Choose message m from space {0,1}
- $C_i = pq+2r+m$
- Evaluation $(P_k, C, CT_1, CT_2,..., CT_t)$ where C is a circuit with AND's and XOR' gates which can construct Adds and Multiplies
  - $CT_{add} = CT1+ CT2 = (pq_1 + 2r + m_1) + (pq + 2r + m_2) \bmod x_0$ is a valid ciphertext
  - $CT_{mult} = CT1 * CT2 = (pq_1 + 2r + m_1) (pq_2 + 2r + m_2) \bmod x_0$ is a valid ciphertext
  - The circuit outputs a ciphertext CT
- Decryption
  - $m^* = (CT \bmod p) \bmod 2$
  - noise must be small enough
- Somewhat homomorphic since it stops after the noise becomes too large

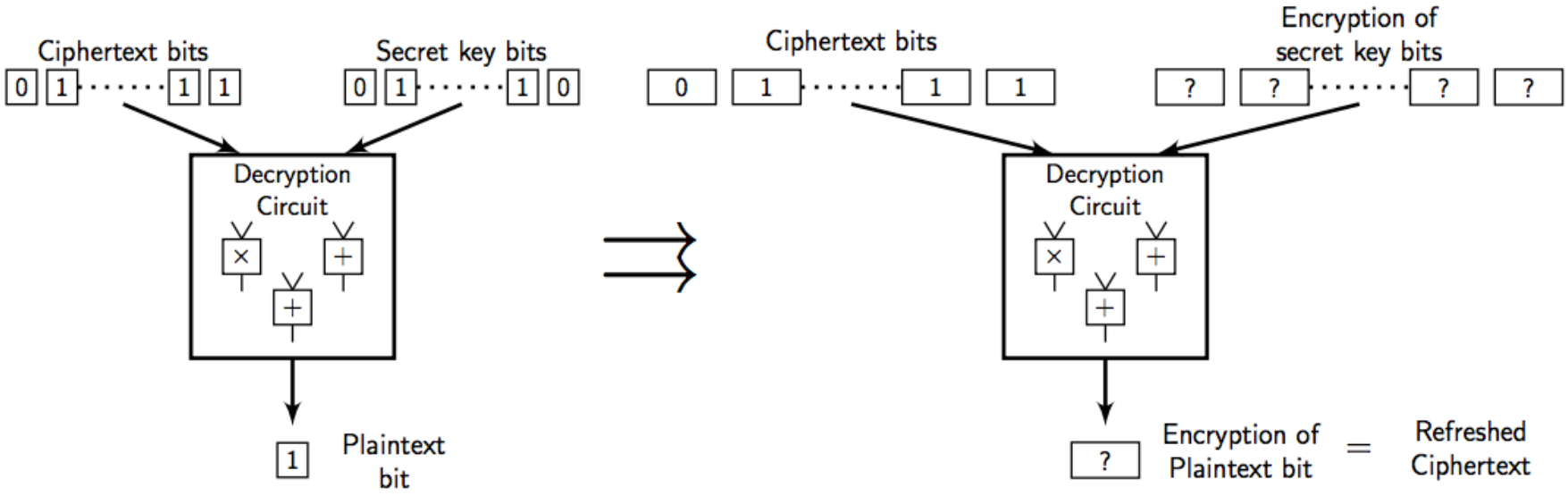
Gentry's Bootstrapping Theorem:

If a SHE scheme is bootstrappable, it can be converted into a FHE scheme.

Bootstrappable means that the SHE can evaluate its own decryption function

If a SHE is not bootstrappable already, it can be made so by "squashing" the decryption circuit.

Note: This is difficult to do. Gentry utilized complex lattice computations to come up with a theoretical but impractical solution.


Ciphertext bits
0 1 ····· 1 1
Secret key bits
0 1 ····· 1 0
Decryption Circuit
×
+
+
1
Plaintext bit
Ciphertext bits
0 1 ····· 1 1
Encryption of secret key bits
? ? ····· ? ?
Decryption Circuit
×
+
+
?
Encryption of Plaintext bit
=
Refreshed Ciphertext

References

C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st ACM Symposium on Theory of Computing – STOC 2009*, pages 169–178. ACM, 2009.

C. Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009. http://crypto.stanford.edu/craig.

C Gentry, et al. Fully Homomorphic Encryption over the Integers. Advances in Cryptology – EUROCRYPT 2010.