Chapter 2
Homomorphic Encryption

Abstract Homomorphic encryption is a form of encryption which allows specific types of computations to be carried out on ciphertexts and generate an encrypted result which, when decrypted, matches the result of operations performed on the plaintexts. This is a desirable feature in modern communication system architectures. RSA is the first public-key encryption scheme with a homomorphic property. However, for security, RSA has to pad a message with random bits before encryption to achieve semantic security. The padding results in RSA losing the homomorphic property. To avoid padding messages, many public-key encryption schemes with various homomorphic properties have been proposed in last three decades. In this chapter, we introduce basic homomorphic encryption techniques. It begins with a formal definition of homomorphic encryption, followed by some well-known homomorphic encryption schemes.

2.1 Homomorphic Encryption Definition

In abstract algebra, a homomorphism is a structure-preserving map between two algebraic structures, such as groups.

A group is a set, G, together with an operation $\circ$ (called the group law of G) that combines any two elements a and b to form another element, denoted $a \circ b$. To qualify as a group, the set and operation, $(G, \circ)$, must satisfy four requirements known as the group axioms:

- Closure: For all a, b in G, the result of the operation, $a \circ b$, is also in G.
- Associativity: For all a, b, and c in G, $(a \circ b) \circ c = a \circ (b \circ c)$.
- Identity element: There exists an element e in G, such that for every element a in G, the equality $e \circ a = a \circ e = a$ holds. Such an element is unique, and thus one speaks of the identity element.
- Inverse element: For each a in G, there exists an element b in G such that $a \circ b = b \circ a = e$, where e is the identity element.

The identity element of a group G is often written as 1.

The result of an operation may depend on the order of the operands. In other words, the result of combining element a with element b need not yield the same result as combining element b with element a; the equation $a \circ b = b \circ a$ may not

© Xun Yi, Russell Paulet, Elisa Bertino 2014

X. Yi et al., *Homomorphic Encryption and Applications*, SpringerBriefs in Computer Science, DOI 10.1007/978-3-319-12229-8_2

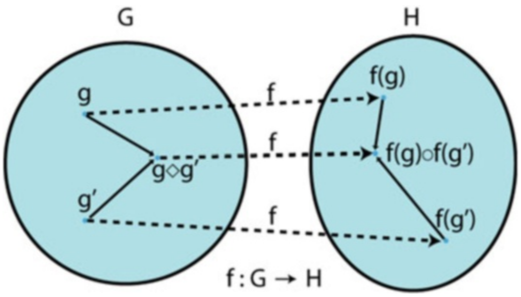

Fig. 2.1 Group Homomorphism

always be true. This equation always holds in the group of integers under addition, because $a + b = b + a$ for any two integers (commutativity of addition). Groups for which the commutativity equation $a \circ b = b \circ a$ always holds are called abelian groups.

Given two groups $(G, \diamond)$ and $(H, \circ)$, a group homomorphism from $(G, \diamond)$ to $(H, \circ)$ is a function $f : G \rightarrow H$ such that for all g and g' in G it holds that

$$f(g \diamond g') = f(g) \circ f(g') \tag{2.1}$$

Group homomorphism can be illustrated as in Fig. 2.1.

Let (P, C, K, E, D) be an encryption scheme, where P, C are the plaintext and ciphertext spaces, K is the key space, and E, D are the encryption and decryption algorithms. Assume that the plaintexts forms a group $(P, \diamond)$ and the ciphertexts forms a group $(C, \circ)$, then the encryption algorithm E is a map from the group P to the group C, i.e., $E_k : P \rightarrow C$, where $k \in K$ is either a secret key (in a secret key cryptosystem) or a public key (in a public-key cryptosystem).

For all a and b in P and k in K, if

$$E_k(a) \circ E_k(b) = E_k(a \diamond b) \tag{2.2}$$

the encryption scheme is **homomorphic**.

In an unpadded RSA [18], assume that the public key $pk = (n, e)$, the plaintexts form a group $(P, \cdot)$, and the ciphertexts form a group $(C, \cdot)$, where $\cdot$ is the modular multiplication. For any two plaintexts m_1, m_2 in P, it holds that

$$\begin{aligned} E(m_1, pk) \cdot E(m_2, pk) &= m_1^e \cdot m_2^e (mod\ n) \\ &= (m_1 \cdot m_2)^e (mod\ n) \\ &= E(m_1 \cdot m_2, pk) \end{aligned}$$

Therefore, the unpadded RSA has the homomorphic property. Unfortunately, the unpadded RSA is insecure.

2.2 Goldwasser–Micali Encryption Scheme

The Goldwasser–Micali (GM) encryption scheme [7] is a public-key encryption algorithm developed by Shafi Goldwasser and Silvio Micali in 1982. GM has the distinction of being the first probabilistic public-key encryption scheme which is provably secure under standard cryptographic assumptions. However, it is not an efficient cryptosystem, as ciphertexts may be several hundred times larger than the initial plaintext. To prove the security properties of the cryptosystem, Goldwasser and Micali proposed the widely used definition of semantic security.

GM consists of three algorithms: a probabilistic key generation algorithm which produces a public and a private key, a probabilistic encryption algorithm, and a deterministic decryption algorithm.

The scheme relies on deciding whether a given value x is a square mod N, given the factorization (p, q) of N. This can be accomplished using the following procedure:

Compute

$$x_p = x(mod\ p) \tag{2.3}$$

$$x_q = x(mod\ q) \tag{2.4}$$

If

$$x_p^{(p-1)/2} = 1(mod\ p) \tag{2.5}$$

$$x_q^{(q-1)/2} = 1(mod\ q) \tag{2.6}$$

then x is a quadratic residue mod N.

Key Generation: The modulus used in GM encryption is generated in the same manner as in the RSA cryptosystem.

Alice generates two distinct large prime numbers p and q, such that $p = q = 3(mod\ 4)$, randomly and independently of each other. Alice computes $N = pq$. She then finds some non-residue a such that

$$a_p^{(p-1)/2} = -1(mod\ p), a_q^{(q-1)/2} = -1(mod\ q)$$

The public key consists of (a, N). The secret key is the factorization (p, q).

Encryption: Suppose Bob wishes to send a message m to Alice. Bob first encodes m as a string of bits $(m_1, \cdots, m_n)$.

For every bit m_i, Bob generates a random value b_i from the group of units modulo N, or $\gcd(b_i, N) = 1$. He outputs the value

$$c_i = b_i^2 \cdot a^{m_i}(mod\ N) \tag{2.7}$$

Bob sends the ciphertext $(c_1, c_2, \cdots, c_n)$ to Alice.

Decryption: Alice receives $(c_1, c_2, \cdots, c_n)$. She can recover m using the following procedure:

For each i, using the prime factorization (p, q), Alice determines whether the value c_i is a quadratic residue; if so, $m_i = 0$, otherwise $m_i = 1$. Alice outputs the message $m = (m_1, \cdots, m_n)$.

GM Example: We choose small parameters in this example. In key generation, we let

$$p = 7, q = 11$$

where $p = q = 3(mod\ 4)$. So

$$N = pq = 77$$

Take

$$a = 6$$

where

$$6^{(7-1)/2} = -1(mod\ 7), 6^{(11-1)/2} = -1(mod\ 11)$$

The public key is (6, 77) and the private key is (7,11).

To encrypt 3-bit message $m_1 m_2 m_3 = 101$. Choose

$$b_1 = 2, b_2 = 3, b_3 = 5$$

and compute

$$c_1 = 2^2 \cdot 6^1 = 24(mod\ 77)$$

$$c_2 = 3^2 \cdot 6^0 = 9(mod\ 77)$$

$$c_3 = 5^2 \cdot 6^1 = 73(mod\ 77)$$

The ciphertext is (24,9,73).

To decrypt the ciphertext, compute

$$24^{(7-1)/2} = -1(mod\ 7)$$

$$9^{(7-1)/2} = 1(mod\ 7), 9^{(11-1)/2} = 1(mod\ 11)$$

$$73^{(7-1)/2} = -1(mod\ 7)$$

This shows that 24 and 73 are non-quadratic residue and 9 is quadratic residue, and thus outputs the plaintext 101.

Homomorphic Property: The GM encryption scheme has a homomorphic property, in the sense that if c_0, c_1 are the encryptions of bits m_0, m_1, then $c_0c_1(mod\ N)$ will be an encryption of $m_0 \oplus m_1$, where $\oplus$ denotes addition modulo 2 (i.e., exclusive-OR).

Assume that

$$c_0 = b_0^2 \cdot a^{m_0}(mod\ N), c_1 = b_1^2 \cdot a^{m_1}(mod\ N)$$

we have

$$\begin{aligned} c_0 \cdot c_1 &= (b_0^2 \cdot a^{m_0}) \cdot (b_1^2 \cdot a^{m_1})(mod\ N) \\ &= (b_0 b_1)^2 \cdot a^{m_0+m_1}(mod\ N) \end{aligned}$$

When $m_0 + m_1$ is either 0 or 1, we have $m_0 + m_1 = m_0 \oplus m_1$. When $m_0 = m_1 = 1$, $m_0 + m_1 = 2$ and $c_0c_1(mod\ N)$ is a quadratic residue and thus it is an encryption of 0. In this case, we have $m_0 \oplus m_1 = 1 \oplus 1 = 0$ as well.

Security: The GM encryption scheme is a probabilistic encryption [8]. Probabilistic encryption refers to the use of randomness in an encryption algorithm, so that when encrypting the same message several times it will, in general, yield different ciphertexts. The term "probabilistic encryption" is typically used in reference to public-key encryption algorithms; however, various secret key encryption algorithms achieve a similar property (e.g., block ciphers when used in a chaining mode such as CBC). To be semantically secure, that is, to hide even partial information about the plaintext, an encryption algorithm must be probabilistic.

Probabilistic encryption is particularly important when using public-key encryption. Suppose that the adversary observes a ciphertext and suspects that the plaintext is either "YES" or "NO." When a deterministic encryption algorithm is used, the adversary can simply try encrypting each of his or her guesses under the recipient's public key and compare each result to the target ciphertext. To combat this attack, public-key encryption schemes must incorporate an element of randomness, ensuring that each plaintext maps into one of a large number of possible ciphertexts.

An intuitive approach to converting a deterministic encryption scheme into a probabilistic one is to simply pad the plaintext with a random string before encrypting with the deterministic algorithm, such as padding RSA. Conversely, decryption involves applying a deterministic algorithm and ignoring the random padding. However, early schemes which applied this naive approach were broken due to limitations in some deterministic encryption schemes. Techniques such as OAEP integrate random padding in a manner that is secure using any trapdoor permutation.

The GM encryption scheme is semantically secure [8]. Semantic security is commonly defined by the following game:

- *Initialize*: The challenger runs the key generation algorithm, gives the public key pk to a probabilistic polynomial time-bounded (PPT) adversary, but keeps the private key sk to itself.
- *Phase* 1: The adversary adaptively asks a number of different encryption queries $C_i = \mathsf{E}(m_i, pk)$ for m_i, where $i = 1, 2, \cdots, n$.
- *Challenge*: Once the adversary decides that Phase 1 is over, it outputs a pair of equal length plaintexts (M_0, M_1) on which it wishes to be challenged. The challenger picks a random bit $b \in \{0, 1\}$ and sends $C = \mathsf{E}(M_b, pk)$ as the challenge to the adversary.
- *Phase* 2: The adversary issues more encryption queries adaptively as in Phase 1.
- *Guess*: Finally, the adversary outputs a guess $b' \in \{0, 1\}$ and wins the game if $b' = b$.

The public-key encryption cryptosystem is semantically secure under chosen-plaintext attack if the adversary cannot determine which of the two messages was chosen by the challenger, with probability significantly greater than 1/2 (the success rate of random guessing).

The GM encryption scheme is semantically secure based on the assumed intractability of the quadratic residuosity problem modulo a composite $N = pq$ where p, q are large primes. This assumption states that given (a, N) it is difficult to determine whether a is a quadratic residue modulo N (i.e., $a = b^2 (mod\ N)$ for some b). The quadratic residue problem is easily solved given the factorization of N. The GM encryption scheme leverages this asymmetry by encrypting individual plaintext bits as either random quadratic residues or non-residues modulo N. Recipients use the factorization of N as a secret key and decrypt the message by testing the quadratic residuosity of the received ciphertext values.

Because the GM encryption scheme produces a value of size approximately $|N|$ to encrypt every single bit of a plaintext, GM encryption results in substantial ciphertext expansion. To prevent factorization attacks, it is recommended that $|N|$ be several hundred bits or more. Thus, the scheme serves mainly as a proof of concept, and more efficient provably secure schemes such as ElGamal encryption scheme have been developed since.

2.3 ElGamal Encryption Scheme

The ElGamal encryption scheme [4] is a public-key encryption algorithm based on the Diffie–Hellman key exchange. It was invented by Taher Elgamal in 1985. The ElGamal encryption scheme is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems. The ElGamal encryption scheme can be defined over any cyclic group G. Its security depends upon the difficulty of a certain problem in G related to computing discrete logarithms.

The ElGamal encryption scheme consists of three components: the key generation, the encryption algorithm, and the decryption algorithm.

Key Generation: The key generator works as follows:

Alice generates an efficient description of a cyclic group G, of order q, with generator g.

Alice chooses a random $x \in \{1, \dots, q-1\}$.

Alice computes

$$y = g^x \tag{2.8}$$

Alice publishes y along with the description of G, q, g, as her public key. Alice retains x, as her private key which must be kept secret.

Encryption: The encryption algorithm works as follows:

To encrypt a message m, to Alice under her public key (G, q, g, y), Bob chooses a random $r \in \{1, \dots, q-1\}$, then computes

$$c_1 = g^r \tag{2.9}$$

Bob computes the shared secret

$$s = y^r \tag{2.10}$$

Bob converts his secret message m, into an element $m' \in G$.

Bob computes

$$c_2 = m' \cdot s \tag{2.11}$$

Bob sends the ciphertext $(c_1, c_2) = (g^r, m' \cdot y^r)$ to Alice.

Note that one can easily find y^r, if one knows m'. Therefore, a new r, is generated for every message to improve security. For this reason, r, is also called an ephemeral key.

Decryption: The decryption algorithm works as follows:

To decrypt a ciphertext (c_1, c_2), with her private key x, Alice computes the shared secret

$$t = c_1^x \tag{2.12}$$

and then computes

$$m' = c_2 \cdot t^{-1} \tag{2.13}$$

which she then converts back into the plaintext message m, where t^{-1} is the inverse of t in the group G (e.g., modular multiplicative inverse if G is a subgroup of a multiplicative group of integers modulo n).

The decryption algorithm produces the intended message, since

$$\begin{aligned} c_2 \cdot t^{-1} &= (m' \cdot s) \cdot c_1^{-x} \\ &= m' \cdot y^r \cdot g^{-xr} \\ &= m' \cdot g^{xr} \cdot g^{-xr} \\ &= m' \end{aligned}$$

The ElGamal encryption scheme is probabilistic, meaning that a single plaintext can be encrypted to many possible ciphertexts, with the consequence that a general ElGamal encryption produces a 2:1 expansion in size from plaintext to ciphertext.

Encryption under ElGamal requires two exponentiations; however, these exponentiations are independent of the message and can be computed ahead of time if need be. Decryption only requires one exponentiation.

The division by t can be avoided by using an alternative method for decryption. To decrypt a ciphertext (c_1, c_2), with Alice's private key x, Alice computes $t' = c_1^{q-x} = g^{(q-x)r}$. t' is the inverse of t. This is a consequence of Lagrange's theorem, because

$$t \cdot t' = g^{xr} \cdot g^{(q-x)r} = (g^q)^r = 1^r = 1$$

where 1 is the identity element of G.

Alice then computes $m' = c_2 \cdot t'$, by which she then converts back into the plaintext message m. The decryption algorithm produces the intended message, since

$$c_2 \cdot t' = m' \cdot s \cdot t' = m' \cdot y^r \cdot t' = m' \cdot g^{xr} \cdot t' = m' \cdot (g^r)^x \cdot t' = m' \cdot c_1^x \cdot t' = m' \cdot t \cdot t' = m'$$

ElGamal Example: An example of the ElGamal encryption with small parameters is given as follows:

At first, Alice generates a prime modulo p and a group generator g which is between 1 and $p - 1$:

$$p = 2879$$
$$g = 2585$$

Alice selects a random number (x) which will be her private key:

$$x = 47$$

She then calculates

$$y = g^x = 2585^{47} = 2826(mod\ 2879)$$

Alice's public key is now (p, g, y) and sends them to Bob. The private key x is known to Alice only.

Bob then creates a message

$$m = 77$$

and then selects a random value

$$r = 65$$

and calculates the ciphertext (c_1, c_2) where

$$c_1 = g^r = 2585^{65} = 319(mod\ 2879)$$
$$c_2 = m \cdot y^r = 77 \cdot 2826^{65} = 472(mod\ 2879)$$

Alice can decrypt the ciphertext:

$$c_2/c_1^x = 472/319^{47} = 77(mod\ 2879).$$

Homomorphic Property: ElGamal encryption scheme has a homomorphic property. Given two encryptions

$$(c_{11}, c_{12}) = (g^{r_1}, m_1 y^{r_1}), (c_{21}, c_{22}) = (g^{r_2}, m_2 y^{r_2})$$

where r_1, r_2 are randomly chosen from $\{1, 2, \cdots, q-1\}$ and $m_1, m_2 \in G$, one can compute

$$\begin{aligned}(c_{11}, c_{12})(c_{21}, c_{22}) &= (c_{11}c_{21}, c_{12}c_{22}) \\ &= (g^{r_1} g^{r_2}, (m_1 y^{r_1})(m_2 y^{r_2})) \\ &= (g^{r_1+r_2}, (m_1 m_2) y^{r_1+r_2})\end{aligned}$$

The resulted ciphertext is an encryption of $m_1 m_2$.

ElGamal Security: The security of the ElGamal scheme depends on the properties of the underlying group G as well as any padding scheme used on the messages.

If the computational Diffie–Hellman assumption (CDH) holds in the underlying cyclic group G, then the ElGamal encryption function is one way. The CDH is the assumption that a certain computational problem within a cyclic group G is hard. Consider a cyclic group G of order q, the CDH assumption states that, given (g, g^a, g^b) for a randomly chosen generator g and random $a, b \in \{0, \cdots, q-1\}$, it is computationally intractable to compute the value g^{ab}.

If the decisional Diffie–Hellman assumption (DDH) holds in G, then ElGamal achieves semantic security. Semantic security is not implied by the CDH alone. The DDH is a computational hardness assumption about a certain problem involving discrete logarithms in cyclic groups. Consider a (multiplicative) cyclic group G of order q, and with generator g. The DDH assumption states that, given g^a and g^b for uniformly and independently chosen $a, b \in \mathbb{Z}_q$, the value g^{ab} "looks like" a random element in G. This intuitive notion is formally stated by saying that the following two probability distributions are computationally indistinguishable:

- (g^a, g^b, g^{ab}), where a and b are randomly and independently chosen from $\mathbb{Z}_q$;
- (g^a, g^b, g^c), where a, b, c are randomly and independently chosen from $\mathbb{Z}_q$.

ElGamal encryption is unconditionally malleable and therefore is not secure under chosen-ciphertext attack. For example, given an encryption (c_1, c_2) of some (possibly unknown) message m, one can easily construct a valid encryption $(c_1, 2c_2)$ of the message $2m$.

To achieve chosen-ciphertext security, the scheme must be further modified, or an appropriate padding scheme must be used. Depending on the modification, the DDH assumption may or may not be necessary.

Other schemes related to ElGamal which achieve security against chosen-ciphertext attacks have also been proposed. The Cramer–Shoup cryptosystem [3] is secure under chosen-ciphertext attack assuming DDH holds for G. Its proof does not use the random oracle model. Another proposed scheme is DHAES [1], whose proof requires an assumption that is weaker than the DDH assumption.

The ElGamal encryption scheme is usually used in a hybrid cryptosystem, i.e., the message itself is encrypted using a symmetric cryptosystem and ElGamal is then used to encrypt the key used for the symmetric cryptosystem. This is because asymmetric cryptosystems like ElGamal are usually slower than symmetric ones for the same level of security, so it is faster to encrypt the symmetric key (which most of the time is quite small if compared to the size of the message) with ElGamal and the message (which can be arbitrarily large) with a symmetric cryptosystem.

2.4 Paillier Encryption Scheme

The Paillier encryption scheme [11], named after and invented by Pascal Paillier in 1999, is a probabilistic public-key algorithm. The problem of computing nth residue classes is believed to be computationally difficult. The decisional composite residuosity assumption is the intractability hypothesis upon which this cryptosystem is based.

The Paillier encryption scheme is composed of key generation, encryption, and decryption algorithms as follows:

Key Generation: 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 equal length.

Compute

$$n = pq, \lambda = lcm(p-1, q-1)$$

where lcm stands for the least common multiple.

Select random integer g where $g \in \mathbb{Z}^*_{n^2}$.

Ensure n divides the order of g by checking the existence of the following modular multiplicative inverse:

$$\mu = (L(g^\lambda (mod\ n^2)))^{-1} (mod\ n) \tag{2.14}$$

where function L is defined as

$$L(u) = \frac{u-1}{n} \tag{2.15}$$

Note that the notation 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.

Finally, the public (encryption) key is (n, g) and 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

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

where $\varphi(n) = (p-1)(q-1)$.

Encryption: Let m be a message to be encrypted where $m \in \mathbb{Z}_n$.

Select random r where $r \in \mathbb{Z}^*_n$

Compute ciphertext as

$$c = g^m \cdot r^n (mod\ n^2) \tag{2.16}$$

Decryption: Let c be the ciphertext to decrypt, where $c \in \mathbb{Z}^*_{n^2}$

Compute the plaintext message as:

$$m = L(c^\lambda (mod\ n^2)) \cdot \mu (mod\ n) \tag{2.17}$$

As the original paper points out, decryption is "essentially one exponentiation modulo n^2."

The Paillier encryption scheme exploits the fact that certain discrete logarithms can be computed easily. For example, by binomial theorem,

$$(1+n)^x = \sum_{k=0}^{x} \binom{x}{k} n^k = 1 + nx + \binom{x}{2} n^2 + \text{higher powers of } n$$

This indicates that

$$(1+n)^x = 1 + nx \pmod{n^2}$$

Therefore, if

$$y = (1+n)^x \mod n^2$$

then

$$x = \frac{y-1}{n} \pmod{n}$$

Thus

$$L((1+n)^x(mod\ n^2)) = x \pmod{n}$$

for any $x \in \mathbb{Z}_n$.

Therefore, when $g = n + 1$, we have

$$\begin{aligned} L(c^{\lambda}(mod\ n^2)) \cdot \mu &= L((g^m r^n)^{\lambda}(mod\ n^2)) \cdot \lambda^{-1} \\ &= L((g^{m\lambda}(mod\ n^2)) \cdot \lambda^{-1} \\ &= \lambda \cdot m \cdot \lambda^{-1} = m(mod\ n) \end{aligned}$$

Paillier Example: An example of the Paillier encryption scheme with small parameters is shown as follows.

For ease of calculations, the example will choose small primes, to create a small n. Let

$$p = 7, q = 11$$

then

$$n = pq = 7 \cdot 11 = 77$$

Next, an integer g must be selected from $\mathbb{Z}_{n^2}^*$, such that the order of g is a multiple of n in $\mathbb{Z}_{n^2}$. If we randomly choose the integer

$$g = 5652$$

then all necessary properties, including the yet to be specified condition, are met, as the order of g is $2310 = 30 \cdot 77$ in $\mathbb{Z}_{n^2}$. Thus, the public key for the example will be

$$(n, g) = (77, 5652)$$

To encrypt a message

$$m = 42$$

where $m \in \mathbb{Z}_n$, choose a random

$$r = 23$$

where r is a nonzero integer and $r \in \mathbb{Z}_n$.

Compute

$$\begin{aligned} c &= g^m r^n (mod\ n^2) \\ &= 5652^{42} \cdot 23^{77} (mod\ 5929) \\ &= 4624 (mod\ 5929) \end{aligned}$$

To decrypt the ciphertext c, compute

$$\lambda = lcm(6, 10) = 30$$

Define $L(u) = (u - 1)/n$, compute

$$\begin{aligned} k &= L(g^\lambda (mod\ n^2)) \\ &= L(5652^{30} (mod\ 5929)) \\ &= L(3928) \\ &= (3928 - 1)/77 \\ &= 3927/77 \\ &= 51 \end{aligned}$$

Compute the inverse of k,

$$\begin{aligned} \mu &= k^{-1} (mod\ n) \\ &= 51^{-1} = 74 (mod\ 77) \end{aligned}$$

Compute

$$\begin{aligned} m &= L(c^\lambda mod n^2) \cdot \mu (mod\ n) \\ &= L(4624^{30} (mod\ 5929)) \cdot 74 (mod\ 77) \\ &= L(4852) \cdot 74 (mod\ 77) \\ &= 42 \end{aligned}$$

Homomorphic Properties: A notable feature of the Paillier scheme is its homomorphic properties. Given two ciphertexts $E(m_1, pk) = g^{m_1} r_1^n (mod\ n^2)$ and $E(m_2, pk) = g^{m_2} r_2^n (mod\ n^2)$, where r_1 and r_2 are randomly chosen from $\mathbb{Z}_n^*$, we have

- Homomorphic Addition of Plaintexts

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

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

  because

$$\begin{aligned} E(m_1, pk) \cdot E(m_2, pk) &= (g^{m_1} r_1^n)(g^{m_2} r_2^n)\ (mod\ n^2) \\ &= g^{m_1+m_2} (r_1 r_2)^n (mod\ n^2) \\ &= E(m_1 + m_2, pk) \end{aligned}$$

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

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

  because

$$\begin{aligned} E(m_1, pk) \cdot g^{m_2} &= (g^{m_1} r_1^n) g^{m_2}\ (mod\ n^2) \\ &= g^{m_1+m_2} r_1^n (mod\ n^2) \\ &= E(m_1 + m_2, pk) \end{aligned}$$

- Homomorphic Multiplication of Plaintexts

  An encrypted plaintext raised to the power of another plaintext will decrypt to the product of the two plaintexts, i.e.,

$$D(E(m_1, pk)^{m_2} (mod\ n^2)) = m_1 m_2 (mod\ n)$$

  because

$$\begin{aligned} E(m_1, pk)^{m_2} &= (g^{m_1} r_1^n)^{m_2}\ (mod\ n^2) \\ &= g^{m_1 m_2} (r_1^{m_2})^n (mod\ n^2) \\ &= E(m_1 m_2, pk) \end{aligned}$$

  More generally, an encrypted plaintext raised to a constant k will decrypt to the product of the plaintext and the constant, i.e.,

$$D(E(m_1, pk)^k (mod\ n^2)) = km_1(mod\ n)$$

However, given the Paillier encryptions of two messages, there is no known way to compute an encryption of the product of these messages without knowing the private key.

Paillier Security: The Paillier encryption scheme provides semantic security against chosen-plaintext attacks (IND-CPA). The ability to successfully distinguish the challenge ciphertext essentially amounts to the ability to decide composite residuosity. The semantic security of the Paillier encryption scheme was proved under the decisional composite residuosity (DCR) assumption—the DCR problem is intractable.

The DCR problem states as follows: Given a composite N and an integer z, it is hard to decide whether z is a N-residue modulo N^2 or not, i.e., whether there exists y such that

$$z = y^n (mod\ n^2)$$

Because of the homomorphic properties, the Paillier encryption scheme, however, is malleable and therefore does not protect against adaptive chosen-ciphertext attacks (IND-CCA2). Usually in cryptography the notion of malleability is not seen as an "advantage," but under certain applications such as secure electronic voting and threshold cryptosystems, this property may indeed be necessary.

Paillier and Pointcheval [12] however went on to propose an improved cryptosystem that incorporates the combined hashing of message m with random r. Similar in intent to the Cramer–Shoup cryptosystem, the hashing prevents an attacker, given only c, from being able to change m in a meaningful way. Through this adaptation the improved scheme can be shown to be IND-CCA2 secure in the random oracle model.

2.5 Boneh–Goh–Nissim Encryption Scheme

Boneh–Goh–Nissim encryption scheme [2], BGN scheme by brevity, resembles the Paillier [11] and the Okamoto–Uchiyama [10] encryption schemes. The BGN scheme was the first to allow both additions and multiplications with a constant-size ciphertext. The multiplication is possible due to the fact that pairings can be defined for elliptic curves.

Let G_1, G_2 be additive groups and G_T a multiplicative group, all of prime order p. Let $P \in G_1$, $Q \in G_2$ be generators of G_1 and G_2, respectively.

A pairing is a map

$$e : G_1 \times G_2 \to G_T$$

for which the following holds:

1. Bilinearity: $\forall a, b \in \mathbb{Z}_p^*$:

$$e(P^a, Q^b) = e(P, Q)^{ab}$$

2. Non-degeneracy: $e(P, Q) \neq 1$.
3. For practical purposes, e has to be computable in an efficient manner.

In cases when $G_1 = G_2 = G$, the pairing is called symmetric. If, furthermore, G is cyclic, the map e will be commutative; that is, for any $P, Q \in G$, we have

$$e(P, Q) = e(Q, P)$$

This is because for a generator $g \in G$, there exist integers p, q such that $P = g^p$ and $Q = g^q$. Therefore

$$e(P, Q) = e(g^p, g^q) = e(g, g)^{pq} = e(g^q, g^p) = e(Q, P)$$

On the basis of pairing, BGN scheme can be described by three algorithms—key generation, encryption, and decryption algorithms—as follows:

Key Generation: Given a security parameter $\lambda \in \mathbb{Z}^+$, generate a tuple (q_1, q_2, G, G_1, e), where q_1 and q_2 are two distinct large primes, G is a cyclic group of order q_1q_2, and e is a pairing map $e : G \times G \to G_1$. Let $N = q_1q_2$. Pick up two random generators g, u from G and set $h = u^{q_2}$. Then h is a random generator of the subgroup of G of order q_1. The public key is $PK = \{N, G, G_1, e, g, h\}$. The private key $SK = q_1$.

Encryption: Assume the message space consists of integers in the set $\{0, 1, \cdots, T\}$ with $T < q_2$. We encrypt bits in which case $T = 1$. To encrypt a message m using the public key PK, pick a random r from $\{1, 2, \cdots, N\}$ and compute

$$C = g^m h^r \in G \tag{2.18}$$

Output C as the ciphertext.

Decryption: To decrypt a ciphertext C using the private key $SK = q_1$, observe that

$$C^{q_1} = (g^m h^r)^{q_1} = (g^{q_1})^m \tag{2.19}$$

To recover the message m, it suffices to compute the discrete logarithm of C^{q_1} to the base g^{q_1}. Since $0 \leq m \leq T$, this takes expected time $O(\sqrt{T})$ using Pollard's lambda method [9].

Homomorphic Properties: The BGN scheme is clearly additively homomorphic. Let $PK = \{N, G, G_1, e, g, h\}$ be a public key. Given two ciphertexts $C_1 = g^{m_1}h^{r_1} \in G, C_2 = g^{m_2}h^{r_2} \in G$ of messages $m_1, m_2 \in \{0, 1, \cdots, T\}$ respectively, anyone can create a uniformly distributed encryption of $m_1 + m_2 (mod\ N)$

by computing the product

$$C = C_1 C_2 h^r \tag{2.20}$$

for a random r in $\{1, 2, \cdots, N-1\}$, because

$$C_1 C_2 h^r = (g^{m_1} h^{r_1})(g^{m_2} h^{r_2}) h^r = g^{m_1+m_2} h^{r_1+r_2+r}$$

is an encryption of $m_1 + m_2$.

More importantly, anyone can multiply two encrypted messages once using the bilinear map. Let

$$g_1 = e(g, g)$$

and

$$h_1 = e(g, h)$$

then g_1 is of order N and h_1 is of order q_1. There is some (unknown) $\alpha \in \mathbb{Z}$ such that

$$h = g^{\alpha q_2}$$

Suppose that we are given two ciphertexts $C_1 = g^{m_1} h^{r_1} \in G$ and $C_2 = g^{m_2} h^{r_2} \in G$. To build an encryption of the product $m_1 m_2 (mod\ N)$, (1) pick a random $r \in \mathbb{Z}_N$, and (2) let

$$C = e(C_1, C_2) h_1^r \in G_1 \tag{2.21}$$

We have

$$\begin{aligned} C &= e(C_1, C_2) h_1^r \\ &= e(g^{m_1} h^{r_1}, g^{m_2} h^{r_2}) h_1^r \\ &= e(g^{m_1+\alpha q_2 r_1}, g^{m_2+\alpha q_2 r_2}) h_1^r \\ &= e(g, g)^{(m_1+\alpha q_2 r_1)(m_2+\alpha q_2 r_2)} h_1^r \\ &= e(g, g)^{m_1 m_2 + \alpha q_2 (m_1 r_2 + m_2 r_1 + \alpha q_2 r_1 r_2)} h_1^r \\ &= e(g, g)^{m_1 m_2} h_1^{r + m_1 r_2 + m_2 r_1 + \alpha q_2 r_1 r_2} \end{aligned}$$

where $r + m_1 r_2 + m_2 r_1 + \alpha q_2 r_1 r_2$ is distributed uniformly in $\mathbb{Z}_N$. Thus C is a uniformly distributed encryption of $m_1 m_2 (mod\ N)$, but in G_1 rather than G. We note that the BGN scheme is still additively homomorphic in G_1.

BGN Example: We will demonstrate the operation of the BGN scheme with a small example. First we choose two distinct prime numbers

$$q_1 = 7, q_2 = 11$$

and compute the product

$$N = q_1 q_2 = 77$$

Next we construct an elliptic curve group with order N that has an associated bilinear map e. The equation for the elliptic curve is

$$y^2 = x^3 + x$$

and is defined over the field F_q for some prime $q = 3 \bmod 4$. In this example, we set

$$q = 307$$

Therefore, the curve is supersingular with $\#(E(q)) = q + 1 = 308$ rational points, which contains a subgroup G with the order $N = 77$ (=308/4).

Within the group G, we choose two random generators

$$g = [182, 240], u = [28, 262]$$

where these two generators have order N, and compute

$$h = u^{q_2} = [28, 262]^{11} = [99, 120]$$

where h has order $q_1 = 7$.

We compute the ciphertext of a message

$$m = 2$$

Take $r = 5$ and compute

$$C = g^m h^r = [182, 240]^2 \oplus [99, 120]^5 = [256, 265]$$

To decrypt we first compute

$$\hat{g} = g^{q_1} = [182, 240]^7 = [146, 60]$$

and

$$C^{q_1} = [256, 265]^7 = [299, 44]$$

Now we find the discrete logarithm by iterating through all the powers of $\hat{g} = g^{q_1}$ as follows:

$$\hat{g}^1 = [146, 60]$$
$$\hat{g}^2 = [299, 44]$$
$$\hat{g}^3 = [272, 206]$$
$$\hat{g}^4 = [191, 151]$$
$$\hat{g}^5 = [79, 171]$$
$$\hat{g}^6 = [79, 136]$$
$$\hat{g}^7 = [191, 156]$$
$$\hat{g}^8 = [272, 101]$$
$$\hat{g}^9 = [299, 263]$$
$$\hat{g}^{10} = [146, 247]$$
$$\hat{g}^{11} = \infty$$

Observe that $\hat{g}^2 = C^{q_1}$. Therefore, decryption of the ciphertext equals 2, which is the same as the original message.

BGN Security: The BGN encryption scheme has been proved to be semantically secure on basis of the subgroup decision problem in [2]. The subgroup decision (SD) problem is stated as follows.

Given a group G of composite order $n = pq$, where p, q are distinct (unknown) primes, and generators $g_p \in G_p$ and $g \in G$, distinguish between whether an element x is a random element of the subgroup G_p or a random element of the full group G.

Gjosteen [6] has undertaken an extensive survey of such problems, which he calls subgroup membership problems. For example, the quadratic residuosity problem is a subgroup membership problem: if we let $N = pq$ be a product of two distinct primes and define the group G to be the group of elements of $\mathbb{Z}_N^*$ with Jacobi symbol 1, the problem is to determine whether a given element in G lies in the subgroup of squares in G.

Boneh, Goh, and Nissim [2] defined their SD problem for pairs of groups (G, G_1) of composite order $N = pq$ for which there exists a nondegenerate bilinear map, or pairing, $e : G \times G \to G_1$. The problem is to determine whether a given element $x \in G$ is in the subgroup of order p. Note that if g generates G, then $e(g, x)$ is a challenge element for the same problem in G_1; thus if the SD problem is infeasible in G, then it is in G_1 as well.

Freeman [5] developed an abstract framework that encompasses the key properties of bilinear groups of composite order that are required to construct secure pairing-based cryptosystems and showed how to use prime-order elliptic curve groups to construct bilinear groups with the same properties. In particular, he defined a generalized version of the subgroup decision problem and give explicit constructions of bilinear groups in which the generalized subgroup decision assumption follows from the decision Diffie–Hellman assumption, the decision linear assumption, and/or related assumptions in prime-order groups.

References

1. M. Abdalla, M. Bellare, P. Rogaway, DHAES: an encryption scheme based on the Diffie–Hellman problem. Submission to IEEE P1363a, 1998. http://www.di.ens.fr/~mabdalla/papers/dhes.pdf
2. D. Boneh, E. Goh, K. Nissim, Evaluating 2-DNF formulas on ciphertexts, in *Proceedings of Theory of Cryptography, TCC'05*, 2005, pp. 325–341
3. R. Cramer, V. Shoup, A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, in *Proceedings of Advances in Cryptology, CRYPTO'98*, 1998, pp. 13–25
4. T. ElGamal, A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**(4), 469–472 (1985)
5. D.M. Freeman, Converting pairing-based cryptosystems from composite-order groups to prime-order groups, in *Proceedings of Advances in Cryptology, EUROCRYPT'10*, 2010, pp. 44–61
6. K. Gjosteen, Subgroup membership problems and public key cryptosystems, Dissertation, Norwegian University of Science and Technology, 2004
7. S. Goldwasser, S. Micali, Probabilistic encryption and how to play mental poker keeping secret all partial information, in *Proceedings of 14th Symposium on Theory of Computing*, 1982, pp. 365–377
8. S. Goldwasser, S. Micali, Probabilistic encryption. J. Comput. Syst. Sci. **28**(2), 270–299 (1984)
9. A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996
10. T. Okamoto, S. Uchiyama, A new public-key cryptosystem as secure as factoring, in *Proceedings of Advances in Cryptology, EUROCRYPT'98*, 1998, pp. 308–318
11. P. Paillier, Public key cryptosystems based on composite degree residue classes, *Proceedings of Advances in Cryptology, EUROCRYPT'99*, 1999, pp. 223–238
12. P. Paillier, D. Pointcheval, Efficient public-key cryptosystems provably secure against active adversaries, in *Proceedings of Advances in Cryptology, ASIACRYPT'99*, 1999, pp. 165–179

http://www.springer.com/978-3-319-12228-1

Homomorphic Encryption and Applications
Yi, X.; Paulet, R.; Bertino, E.
2014, XII, 126 p. 23 illus., 14 illus. in color., Softcover
ISBN: 978-3-319-12228-1