Paillier cryptosystem

The **Paillier crypto system**, invented by and named after Pascal Paillier in 1999, is a probabilistic asymmetric algorithm for public key cryptography. The problem of computing *n*-th residue classes is believed to be computationally difficult. The decisional composite residuosity assumption is the intractability hypothesis upon which this cryptosystem is based.

The scheme is an additive homomorphic cryptosystem; this means that, given only the public key and the encryption of m_1 and m_2, one can compute the encryption of $m_1 + m_2$.

Algorithm

The scheme works as follows:

Key generation

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 equal length.[1]

2. Compute $n = pq$ and $\lambda = \operatorname{lcm}(p-1, q-1)$. lcm means Least Common Multiple.
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: $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$,

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

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 $\mu = \varphi(n)^{-1} \bmod n$, where $\varphi(n) = (p - 1)(q - 1)$.[1]

Encryption

...

1. Let m be a message to be encrypted where $0 \leq m < n$
2. Select random r where $0 < r < n$ and $r \in \mathbb{Z}_n^*$ (i.e., ensure $gcd(r, n) = 1$)
3. Compute ciphertext as: $c = g^m \cdot r^n \bmod n^2$

Decryption

...

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

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

Homomorphic properties

A notable feature of the Paillier cryptosystem is its homomorphic properties along with its non-deterministic encryption (see Electronic voting in Applications for usage). As the encryption function is additively homomorphic, the following identities can be described:

- **Homomorphic addition of plaintexts** The product of two ciphertexts will decrypt to the sum of their corresponding plaintexts,

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

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

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

- **Homomorphic multiplication of plaintexts**

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

  $$D(E(m_1, r_1)^{m_2} \bmod n^2) = m_1 m_2 \bmod n,$$

  $$D(E(m_2, r_2)^{m_1} \bmod n^2) = m_1 m_2 \bmod n.$$

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

  $$D(E(m_1, r_1)^k \bmod n^2) = k m_1 \bmod n.$$

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.

Background

Paillier cryptosystem 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:

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

Therefore, if:

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

then

$$x \equiv \frac{y-1}{n} \pmod{n}.$$

Thus:

$$L((1+n)^x \bmod n^2) \equiv x \pmod{n}$$

,

where function L is defined as $L(u) = \frac{u-1}{n}$ (quotient of integer division) and $x \in \mathbb{Z}_n$.

Semantic security

...

The original cryptosystem as shown above does provide 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 so-called decisional composite residuosity assumption (DCRA) is believed to be intractable.

Because of the aforementioned homomorphic properties however, the system is malleable, and therefore does not enjoy the highest echelon of semantic security that protects 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 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.

Applications

Electronic voting

Semantic security is not the only consideration. There are situations under which malleability may be desirable. The above homomorphic properties can be utilized by secure electronic voting systems. Consider a simple binary ("for" or "against") vote. Let m voters cast a vote of either *1* (for) or *0* (against). Each voter encrypts their choice before casting their vote. The election official takes the product of the m encrypted votes and then decrypts the result and obtains the value n, which is the sum of all the votes. The election official then knows that n people voted *for* and $m-n$ people voted *against*. The role of the random r ensures that two equivalent votes will encrypt to the same value only with negligible likelihood, hence ensuring voter privacy.

Electronic cash

...

Another feature named in paper is the notion of self-blinding. This is the ability to change one ciphertext into another without changing the content of its decryption. This has application to the development of ecash, an effort originally spearheaded by David Chaum. Imagine paying for an item online without the vendor needing to know your credit card number, and hence your identity. The goal in both electronic cash and electronic voting, is to ensure the e-coin (likewise e-vote) is valid, while at the same time not disclosing the identity of the person with whom it is currently associated.

See also

- The Naccache–Stern cryptosystem and the Okamoto–Uchiyama cryptosystem are historical antecedents of Paillier.

- The Damgård–Jurik cryptosystem is a generalization of Paillier.

References

- Paillier, Pascal (1999). "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". *EUROCRYPT*. Springer. pp. 223–238. doi:10.1007/3-540-48910-X_16 .
- Paillier, Pascal; Pointcheval, David (1999). "Efficient Public-Key Cryptosystems Provably Secure Against Active Adversaries". *ASIACRYPT*. Springer. pp. 165–179. doi:10.1007/978-3-540-48000-6_14 .

- Paillier, Pascal (1999). *Cryptosystems Based on Composite Residuosity* (Ph.D. thesis). École Nationale Supérieure des Télécommunications.
- Paillier, Pascal (2002). "Composite-Residuosity Based Cryptography: An Overview" (PDF). *CryptoBytes*. **5** (1). Archived from the original (PDF) on October 20, 2006.

Notes

1. *Jonathan Katz, Yehuda Lindell, "Introduction to Modern Cryptography: Principles and Protocols," Chapman & Hall/CRC, 2007*

External links

- The Homomorphic Encryption Project implements the Paillier cryptosystem along with its homomorphic operations.

- Encounter: an open-source library providing an implementation of Paillier cryptosystem and a cryptographic counters construction based on the same.
- python-paillier a library for Partially Homomorphic Encryption in Python, including full support for floating point numbers.
- The Paillier cryptosystem interactive simulator demonstrates a voting application.
- An interactive demo of the Paillier cryptosystem.
- A proof-of-concept Javascript implementation of the Paillier cryptosystem with an interactive demo .

- A googletechtalk video on voting using cryptographic methods.
- A Ruby implementation of Paillier homomorphic addition and a zero-knowledge proof protocol (documentation)
