Paillier and Damgård-Jurik cryptosystems

January 31, 2014

Section 1

Paillier cryptosystem [Pai99]

Key Generation I

- Choose two large primes p, q randomly and independently such that $gcd(pq, (p-1)(q-1)) = 1$.
- It can be proved that this property is easily attained, if p, q are of the same length η
- Calculate RSA modulus $n = pq$
- Calculate Carmichael's Function $\boldsymbol{\lambda} = lcm(p-1, q-1) = \frac{(p-1)(q-1)}{gcd(p-1,q-1)}$.
- This function is easy to calculate if we know p, q and has the following very important properties:
  - $\forall x \in \mathbb{Z}_{n^2}^* : x^{\lambda(n)} = 1 \pmod{n}$ because of Carmichael's Theorem
  - $\forall x \in \mathbb{Z}_{n^2}^* : x^{n\lambda(n)} = 1 \pmod{n^2}$ because $n\lambda(n) = \lambda(n^2)$
- Select generator $g \in \mathbb{Z}_{n^2}^*$ such that the order of g is a non zero multiple of n.
- Calculate inverse $\boldsymbol{\mu} = L(g^{\lambda} \pmod{n^2})^{-1} \pmod{n}$ where $L(x) = \frac{x-1}{n}$.

Key Generation II

- This function is very important in the Paillier cryptosystem. Its role can be summarized as:
    - $L()$ is given elements that are equal to $1 \pmod n$
    - $L()$ *'solves'* the discrete log problem and *'decrypts'*
    - The inverse to be calculated, always exists if g is a valid generator
- Release the keys. The public key is (n, g) and the private key (λ, μ)

Encryption and Decryption

Encryption: $\mathbb{Z}_n \times \mathbb{Z}_n^* \to \mathbb{Z}_{n^2}^*$

- Encode message m into $\mathbb{Z}_n$
- Select random $r \in \mathbb{Z}_n^*$
- Return $c = Enc(m, r) = g^m r^n \pmod{n^2}$

Decryption: $\mathbb{Z}_{n^2}^* \to \mathbb{Z}_n$

- Ciphertext $c \in \mathbb{Z}_{n^2}^*$
- Return $m = L(c^\lambda \, mod \, n^2)\mu \pmod{n} = \frac{L(c^\lambda \, mod \, n^2)}{L(g^\lambda \, mod \, n^2)} (modn)$

Security

- The security of the Paillier cryptosystem depends on the **composite residuosity problem** - a generalisation of quadratic residuosity
- Informally: distinguish a random element of $\mathbb{Z}^*_{n^2}$ from an $n-residue$
- More formally:Given $n = pq$ and $z \in \mathbb{Z}^*_{n^2}$ decide if z is $n-residue$ modulo n^2, does there exist $y \in \mathbb{Z}^*_{n^2}$ st: $z = y^n \pmod{n^2}$.

Decisional Composite Residuosity Assumption-DCRA

There is no probabilistic polynomial time algorithm to decide the composite residuosity problem

Observe that if there was an algorithm to check if $z \in \mathbb{Z}^*_{n^2}$ is the encryption of message 0 then we could solve the composite residuosity problem.

Proving Correctness I

Theorem

For $c = Enc(m,r) = g^m r^n \pmod{n^2}$ the decryption operation $\frac{L(c^\lambda \pmod{n^2})}{L(g^\lambda \pmod{n^2})} \pmod{n}$ yields m.

Helpers

- **Fact:** $g = n+1$ can always be used. Proof in [DJ01]
- Notation: $[c]_{n+1}$ refers to the plaintext that corresponds to ciphertext c using $g = n+1$
- This means that: $w = Enc_{n+1}([w]_{n+1}, r)$

Proving Correctness II

Roadmap

1. $\forall x \in \mathbb{Z}_n : (1+n)^x = 1 + nx \pmod{n^2}$
2. $\forall c \in \mathbb{Z}^*_{n^2}$, and proper generators g_1, g_2 : $[c]_{g_1} = [c]_{g_2}[g_2]_{g_1}$
3. $\forall w \in \mathbb{Z}^*_{n^2} : \boldsymbol{L(w^\lambda \pmod{n^2}) = \lambda[w]_{n+1} \pmod{n}}$

Step 1

Lemma

$\forall x \in \mathbb{Z}_n : (1+n)^x = 1 + nx \pmod{n^2}$

Proof

Most terms of the binomial expansion are zero in $\mathbb{Z}_{n^2}$:
$(1+n)^x \pmod{n^2} = 1 + \binom{x}{1} n + \binom{x}{2} n^2 + \cdots + \binom{x}{x} n^x \pmod{n^2} = 1 + xn \pmod{n^2}$

Step 2

Lemma

$\forall c \in \mathbb{Z}^*_{n^2}$, and proper generators g_1, g_2 : $\quad [c]_{g_1} = [c]_{g_2}[g_2]_{g_1}$

Proof

Let $y = [g_2]_{g_1}$. This means that: $g_2 = g_1^y b^n$ where $b \in_R \mathbb{Z}^*_n$
Let $z = [c]_{g_2}$. This means that $c = g_2^z d^n$ where $d \in_R \mathbb{Z}^*_n$.
Now $c = g_2^z d^n = (g_1^y b^n)^z d^n = g_1^{zy}(b^z d)^n$ which means that $yz = [c]_{g_1}$
Rewriting gives us $[c]_{g_1} = [c]_{g_2}[g_2]_{g_1}$

Step 3: I

Main Lemma

$\forall w \in \mathbb{Z}^*_{n^2} : \boldsymbol{L(w^\lambda \, mod \, n^2) = \lambda[w]_{n+1} \, mod \, n}$

Since $n+1$ is a proper g, $\forall w \in \mathbb{Z}^*_{n^2}$: by definition:

$$w = Enc_{n+1}([w]_{n+1}, r) = (n+1)^{[w]_{n+1}} r^n \pmod{n^2}$$

Now decryption computes:

$$w^\lambda = (n+1)^{\lambda [w]_{n+1}} r^{n\lambda} \pmod{n^2}$$

Because of the binomial terms disappearing $\pmod{n^2}$ we get:

$$w^\lambda = (1 + \lambda [w]_{n+1} n) r^{n\lambda} \pmod{n^2}$$

Since $n\lambda(n) = \lambda(n^2)$:

Step 3: II

$$w^{\lambda} = (1 + \lambda[w]_{n+1} n) r^{\lambda(n^2)} \pmod{n^2}$$

which leaves:

$$w^{\lambda} = (1 + \lambda[w]_{n+1} n)$$

Then: $L(w^{\lambda}) = \frac{w^{\lambda}-1}{n} = \lambda[w]_{n+1}$
So during the decryption operation:

$$\frac{L(c^{\lambda} \bmod n^2)}{L(g^{\lambda} \bmod n^2)} =$$

$$\frac{\lambda[c]_{n+1}}{\lambda[g]_{n+1}} =$$

$$\frac{[c]_{n+1}}{[g]_{n+1}} = \frac{[c]_g [g]_{n+1}}{[g]_{n+1}} = [c]_g = m$$

Section 2

The Damgård-Jurik cryptosystem [DJ01]

Key Generation I

For each $s \geq 1$ a cryptosystem $\mathcal{CS}_s$ and g=$n+1$ a generalisation and simplification can be defined s can be selected at any point in time before encryption, as long as $m < n^s$

- Public key is $n = pq$
- Private key is $\lambda = lcm(p-1, q-1)$
- **Output:** Public key is n and private key is λ

Encryption $\mathbb{Z}_{n^s} \times \mathbb{Z}_n^* \to \mathbb{Z}_{n^{s+1}}^*$

- Choose s that the message m can be encoded into $\mathbb{Z}_{n^s}$
- Select random $r \in \mathbb{Z}_n^*$
- Return $c = Enc(m, r) = (1+n)^m r^{n^s} \pmod{n^{s+1}}$

Decryption $\mathbb{Z}^*_{n^{s+1}} \to \mathbb{Z}_{n^s}$ I

- Ciphertext $c \in \mathbb{Z}^*_{n^{s+1}}$. Discover s by the length of c
- Calculate $c^{\lambda} \pmod{n^{s+1}} = (1+n)^{m\lambda} \pmod{n^s} r^{\lambda n^s} \pmod{n^{s+1}} = (1+n)^{m\lambda} \pmod{n^{s+1}}$ because $r^{\lambda n^s} = r^{\lambda n^{s+1}} = 1 \pmod{n^{s+1}}$
- Compute $m\lambda$ from $(1+n)^{m\lambda} \pmod{n^{s+1}}$. In order to do this we shall use the $L()$ function $\pmod{n^s}$, taking into account that some terms will be reduced to $0 \pmod{n^s}$

$$L((1+n)^{m\lambda} \pmod{n^{s+1}}) =$$
$$\frac{1-(1+n)^{m\lambda} \pmod{n^{s+1}}}{n} \pmod{n^s} =$$
$$m\lambda + \binom{m\lambda}{2} n + \cdots + \binom{m\lambda}{s} n^{s-1} \pmod{n^s}$$

Decryption $\mathbb{Z}^*_{n^{s+1}} \to \mathbb{Z}_{n^s}$ II

Message Extraction

- We cannot extract $m\lambda$ as easily as in the case of n^2
- **Solution**: Extract it step-by-step. Notation switch: extracting i
- First extract the value of $i \pmod n$
- The extract $i \pmod{n^2}$ and so on ...
- Compare to accumulating the digits of a number from the LSB to the MSB (first the final one, then the final two etc)
- If we have computed i_{j-1} then $i_j = i_{j-1} + kn^{j-1}$ where $k \in \{0, \cdots n-1\}$

Decryption $\mathbb{Z}^*_{n^{s+1}} \to \mathbb{Z}_{n^s}$ III

$$\begin{aligned} L((1+n)^i \pmod{n^{j+1}}) &= \\ i_j + \binom{i_j}{2} n + \cdots + \binom{i_j}{j} n^{j-1} \pmod{n^j} &= \\ i_{j-1} + kn^{j-1} + \binom{i_{j-1}}{2} n + \cdots + \binom{i_{j-1}}{j} \pmod{n^j} \end{aligned}$$

The crucial step is the replacement: $\binom{i_j}{t} n^{t-1} = \binom{i_{j-1}}{t} n^{t-1}$ which holds because:

Decryption $\mathbb{Z}^*_{n^{s+1}} \to \mathbb{Z}_{n^s}$ IV

$$\binom{i_j}{t} n^{t-1} = n^{t-1} \prod_{x=1}^{t} \frac{i_j - (t-x)}{x} =$$

$$\frac{n^t}{n} \prod_{x=1}^{t} \frac{i_j - (t-x)}{x} = \frac{1}{n} \prod_{x=1}^{t} \frac{n(i_j - (t-x))}{x} =$$

$$\frac{1}{n} \prod_{x=1}^{t} \frac{n(i_{j-1} + kn^{j-1} - (t-x))}{x} =$$

$$\frac{1}{n} \prod_{x=1}^{t} \frac{(ni_{j-1} + kn^{j} - n(t-x))}{x} = \frac{1}{n} \prod_{x=1}^{t} \frac{(ni_{j-1} - n(t-x))}{x} =$$

$$\frac{n^t}{n} \prod_{x=1}^{t} \frac{(i_{j-1} - (t-x))}{x} = n^{t-1} \binom{i_{j-1}}{t}$$

Decryption $\mathbb{Z}^*_{n^{s+1}} \rightarrow \mathbb{Z}_{n^s}$ V

- As a result:
  $L((1+n)^i \pmod{n^{j+1}}) = kn^{j-1} + i_{j-1} + \binom{i_{j-1}}{2}n + \cdots + \binom{i_{j-1}}{j} \pmod{n^j}$
- which means that:$kn^{j-1} = L((1+n)^i \pmod{n^{j+1}}) - (\binom{i_{j-1}}{2}n + \cdots + \binom{i_{j-1}}{j}) \pmod{n^j}$
- If we replace kn^{j-1} in $i_j = i_{j-1} + kn^{j-1}$ we get:

$$i_j = L((1+n)^i \pmod{n^{j+1}}) - (\binom{i_{j-1}}{2}n + \cdots + \binom{i_{j-1}}{j}n^{j-1}) \pmod{n^j}$$

- After extraction we can retrieve m from $m\lambda$ by multiplying with $\lambda{-1} \pmod{n^{s+1}}$

Theorem

$\forall s$ the simplified version is one way if Paillier ($\mathcal{CS}_1$) is one way and semantically secure iff the DCRA is true.

Extraction algorithm in Python

```
i=0
for j in range(1,s+1):
    nj = n**j
    t1=(x%(nj*n)-1)//n
    t2=i
    sum=0
    for k in range(2,j+1):
        sum += binomial(t2,k)*n**(k-1)%nj
    t1=(t1-sum)%nj
    i=t1
```

References I

[DJ01] Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, PKC '01, pages 119–136, London, UK, UK, 2001. Springer-Verlag.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.