A Generalization of Paillier's Public-Key System With Fast Decryption

Ying Guo[1], Zhenfu Cao (✉)[2,3], and Xiaolei Dong[2]

[1]Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai, China
sjtuguoying@126.com

[2]Shanghai Key Laboratory of Trustworthy Computing,
East China Normal University, Shanghai, China
zfcao@sei.ecnu.edu.cn, dongxiaolei@sei.ecnu.edu.cn

[3]Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen and
Shanghai Institute of Intelligent Science and Technology, Tongji University, China

Abstract

In Paillier's scheme, $c = y^m x^n \bmod n^2$, $m \in Z_n$, $x \in Z_{n^2}^*$, $n = PQ$ is a product of two large primes. Damgård and Jurik generalized Paillier's scheme to reduce the ciphertext expansion, $c = y^m x^{n^s} \bmod n^{s+1}$, $m \in Z_{n^s}$, $x \in Z_{n^{s+1}}^*$. In this paper, we propose a new generalization of Paillier's scheme and prove that our scheme is IND-CPA secure under k-subgroup assumption for Π_k. Compared to Damgård and Jurik's generalization, our scheme has three advantages. (a)We use the modulus P^aQ^b instead of P^aQ^a, so it is more general. (b)We use a general y satisfying $P^{a-1}|order_{P^a}(y)$, $Q^{b-1}|order_{Q^b}(y)$ instead of $y = (1+PQ)^j x \bmod N$ which is used in Damgård and Jurik's generalization. (c)Our decryption scheme is more efficient than Damgård and Jurik's generalization system.

Keywords: Paillier's PKC scheme, Damgård and Jurik's generalization scheme, discrete logarithm problem, k-subgroup assumption for Π_k

1 Introduction

Paillier proposed a public key encryption scheme Pai99 [1] in 1999. Pai99 is efficient and homomorphic, which means $E_{pk}(m_1)E_{pk}(m_2) = E_{pk}(m_1 + m_2)$ such that it is widely used in many applications [2–14]. In Pai99, $N = P^2Q^2, P, Q$ are large primes. The message space $M = \{m|m < k\}$, $k = PQ$. Randomly choose y such that $ord_N(y^{[P-1,Q-1]}) = k$. The public key $pk = (N, y, k)$ and the secret key $sk = [P - 1, Q - 1]$. The encryption scheme $c = y^m x^k \bmod N$, $m \in M$, $x \in Z_N^*$. The decryption scheme $m = L(c^{[P-1,Q-1]} \bmod N)/L(y^{[P-1,Q-1]} \bmod N) \bmod k$, $L(x) = (x-1)/k$. Pai99 is IND-CPA secure under (N, PQ) residuosity assumption and the ciphertext expansion $r = \dfrac{|c|}{|m|} = \dfrac{|N|}{|k|} = 2$.

In order to reduce the ciphertext expansion of Pai99, Ivan Damgård and Mads Jurik proposed a generalisation of Pai99 [15]. In DJ01, $N = P^aQ^a$, $k = P^{a-1}Q^{a-1}$, $y = (1 + PQ)^j x \bmod N$, $(j, PQ) = 1$, $x^{[P-1,Q-1]} \equiv 1 \bmod N$. The encryption scheme is the same as Pai99 but the decryption scheme is more complicated. In decryption scheme, $c^{[P-1,Q-1]} = (1 + PQ)^{mj[P-1,Q-1]} \bmod N$. Computing $\langle m \rangle_k$ from $(1+PQ)^{mj[P-1,Q-1]} \bmod N$, they use binomial expansion and mathematical induction. We show the method in Section 2.1. DJ01 is IND-CPA secure under (N, PQ) Residuosity assumption and the ciphertext expansion $r = \dfrac{|N|}{|k|} = \frac{a}{a-1}$. Pai99 is a special case of DJ01 when $a = 2$.

Low decryption efficiency is the main problem of DJ01. Another disadvantage is the choice of the public key y. DJ01 required $y = (1 + PQ)^j x \bmod N$ instead of $ord_N(y^{[P-1,Q-1]}) = PQ$ in Pai99 for the sake of specific decryption scheme. In 2008, Obi, Ali and Stipidis proposed a new decryption method [16] for DJ01 and use a general $y \in Z_N^*$ with the order of y a multiple of $P^{a-1}Q^{a-1}$. The decryption scheme is similar to Pai99 in form and we show the method in Section 2.2. When $a = 2$, the decryption scheme is the same as Pai99. OAS08 is the first scheme to use general y in generalisation of Paillier's scheme. The decryption method is more efficient than DJ01. The ciphertext expansion and security of OAS08 is the same as DJ01.

OAS08 is the best work of the generalisation of Paillier's scheme as we know.

In this paper, we give a new generalisation of Pai99. In our scheme, we use $N = P^aQ^b$ instead of $N = P^aQ^a$ so that it is more general. The message space $M = \{m|m < k\}$, $k = P^{k_a}Q^{k_b}$, $0 \leq k_a < a$, $0 \leq k_b < b$. The length of k is l. The ciphertext expansion of our scheme is $r = \frac{a+b}{k_a+k_b}$. If $ak_b = bk_a$, we can public k, otherwise we use N in encryption scheme and public l to ensure $|m| < l$. We also use a general y satisfying $P^{a-1}|order_{P^a}(y)$, $Q^{b-1}|order_{Q^b}(y)$. In our decryption scheme, we compute $\langle m\rangle_{P^{k_a}}$ and $\langle m\rangle_{Q^{k_b}}$ respectively, then we compute $\langle m\rangle_k$ using Chinese Remainder Theorem(CRT). Because of $k|\varphi(N)$, our decryption scheme can be transformed into a discrete logarithm problem that $c^{\varphi(N)/k} \equiv (y^{\varphi(N)/k})^m \bmod N$. We see how to solve this problem under certain conditionsin in Section 2.4. The number of multiplications for decryption of DJ01, OAS08 and our scheme are $O(a\lambda+a^2)$, $O(a+\lambda)$ and $O(\lambda)$ respectively. λ is the security parameter. The speed of our decryption algorithm is independent of a, so our decryption scheme is more efficient than DJ01 and OAS08.

The idea of $N = P^aQ^b$ comes from Okamoto and Uchiyama's scheme [17]. In OU98, $N = P^2Q$, $k = P$, $ord_{P^2}(y^{P-1}) = P$, $|P| = l$. The public key $pk = (N, y, l)$ and $|m| < l$. The encryption scheme $c = y^m x^N \bmod N$, $|m| < l$, $x \in Z_N^*$. The decryption scheme $m = L(c^{P-1} \bmod P^2)/L(y^{P-1} \bmod P^2) \bmod P$, $L(x) = (x-1)/k$. The ciphertext expansion $r = 3$ and OU98 is IND-CPA secure under P-subgroup assumption. According to P-subgroup assumption, we define k-subgroup assumption for Π_k and prove that k-subgroup assumption for Π_k is equal to (N, k) residuosity assumption in Section 2.6. We give the security proof in Section 4 for a special case and the general case. For a special case, our scheme is IND-CPA secure under (k'^2, k') residuosity assumption where $k' = P^{\frac{k_a}{(k_a,k_b)}}Q^{\frac{k_b}{(k_a,k_b)}}$. Both Pai99 and DJ01 satisfy the special case with $k' = PQ$, so we can prove that Pai99 and DJ01 is IND-CPA secure under (P^2Q^2, PQ) residuosity assumption. The same results are proved in their schemes. For the general case, we prove that our scheme is IND-CPA secure under (N, k) residuosity assumption. k can not be public when $ak_b \neq bk_a$, so we use k-subgroup assumption for Π_k instead of (N, k) residuosity assumption in the security proof. Our scheme is IND-CPA secure under k-subgroup assumption

for Π_k.

2 Preliminaries

Let us see some notations (see Table 1), definitions, theorems and assumptions related to our scheme.

Table1: Notations used in this paper

Notation	**Description**
(x, y)	the greatest common divisor of x and y
$[x, y]$	the least common multiple of x and y
$\langle x \rangle_N$	$x \bmod N$, result in a non-negative minimal residue
$\lfloor x \rfloor$	the largest integer less than or equal to x
Z_N	$Z_N = \{0, 1, 2, ..., N-1\}$
Z_N^*	$Z_N^* = \{x \in Z_N \mid (x, N) = 1\}$
$\varphi(N)$	Euler function of N
$ord_N(x)$	the order of x w.r.t. modulus N
$R(N, k)$	$R(N, k) = \{y \mid y = x^k \bmod N,\ x \in Z_N^*\}$
$x \xleftarrow{R} A$	randomly choose x from the set A
$A \backslash B$	$A \backslash B = \{x \mid x \in A,\ x \notin B\}$
$\lvert x \rvert$	the length of the binary string x
$\lvert A \rvert$	the number of elements of the set A
λ	the security parameter of the our scheme
k	the message space $M = \{m \mid m < k\}$
r	ciphertext expansion $r = \dfrac{\lvert c \rvert}{\lvert m \rvert}$

2.1 Decryption Method of DJ01

In this section, we review the main idea of decryption scheme in DJ01. Readers can refer to [15] for more details. The most important step is computing $\langle i \rangle_{n^{a-1}}$ from $c = (1+n)^i \bmod n^a$, where $n = pq$. They give the following algorithm:

$i := 0$;

for $j := 1$ to $a - 1$ do

begin

$t_1 := L(c \bmod n^{j+1})$;

$t_2 := i$;

for $k := 2$ to j do

begin

$i := i - 1$;

$t_2 := t_2 * i \bmod n^j$;

$t_1 := t_1 - \frac{t_2 * n^{k-1}}{k!} \bmod n^j$;

end

$i := t_1$;

end

From the algorithm, we can see the number of multiplications for decryption is $O(a\lambda + a^2)$.

2.2 Decryption Method of OAS08

In this section, we review the decryption scheme of DJ01. Readers can refer to [16] for more details.

In Pai99, $L(x) = (x-1)/n$, $n = PQ$, $x \equiv 1 \bmod n$.

In OAS08, $n = PQ$, $N = n^a$, $k = n^{a-1}$, $y \in Z_N^*$ with the order of y a multiple of k.

Define a new function $L_a(x) = \Sigma_{i=1}^{a}(-1)^{i-1}(x-1)^i/i$. When $a = 1$, $L_a(x)/n = L(x)$.

$d \in Z_n^*$ such that $d \equiv 0 \bmod [P-1, Q-1]$.

The decryption scheme $m = \dfrac{L_{a-1}(c^d \bmod N) \bmod N}{L_{a-1}(y^d \bmod N) \bmod N} \bmod k$.

From the algorithm, we can see the number of multiplications for decryption is $O(a + \lambda)$.

2.3 λ-acceptable Tuple (N, k)

Definition 2.1 (λ-acceptable tuple (N, k))**.**

We call a tuple (N, k) is λ-acceptable if N and k satisfies the following conditions:

P, Q are large primes and $|P| = |Q| = \lambda$. Both $P-1, Q-1$ contain at least one large prime factor.

$N = P^a Q^b$, $a \geq 1$, $b \geq 1$, $k = P^{k_a} Q^{k_b}$, $0 \leq k_a < a$, $0 \leq k_b < b$.

2.4 (y, k, N) Discrete Logarithm Problem

In this section, we introduce (y, k, N) discrete logarithm problem and show that it is easy to solve for a special case which we used in our decryption scheme.

Definition 2.2 ((y, k, N) discrete logarithm problem)**.**

N is positive integer, $y \in Z_N^*$, $ord_N(y) = k$.

$c \equiv y^m \bmod N$. Given (c, y, N, k), compute $< m >_k$.

Theorem 2.1.

If $k^2 | N$, then (y, k, N) discrete logarithm problem can be solved effectively.

Proof.

$d = N/k$, $S = \{x | x \equiv 1 \bmod d,\ 0 < x < N,\ (x, N) = 1\}$, $|S| = k$.

$S' = \{x | x^k \equiv 1 \bmod N,\ 0 < x < N,\ (x, N) = 1\}$, $|S'| = k$.

It is easy to prove that $S = S'$.

$ord_N(y) = k$, $y \in S$, $y = td + 1$, $0 < t < k$.

Define the function $L(x) = \frac{x-1}{d}$, $x \in S$.

$k^2 | N$, $N | d^2$, $y^m = (td + 1)^m \equiv mtd + 1 \equiv c \bmod N$.

$mtd + 1 \equiv c \bmod d$, $c \equiv 1 \bmod d$, $c = t'd + 1$, $c \in S$.

$mtd \equiv t'd \bmod N$, $mt \equiv t' \bmod k$.

$g = (t, k)$, $y^{\frac{k}{g}} = (td + 1)^{\frac{k}{g}} \equiv 1 \bmod N$.

$ord_N(y) | \frac{k}{g}$, $ord_N(y) = k$, $k | \frac{k}{g}$, $g = 1$.

$(t, k) = 1$, $m \equiv t' t^{-1} \equiv \frac{L(c)}{L(y)} \bmod k$. □

2.5 (N, k) Residuosity Assumption

Definition 2.3 ((N, k) residuosity assumption)**.**

(N, k) is λ-acceptable.

$R(N, k) = \{y | y = x^k \bmod N,\ x \in Z_N^*\}$

$D(N, k, x)$ is a distinguisher to decide whether $x \in R(N, k)$.

$$Pr = \left| Pr[D(N, k, x) = 1 | x \xleftarrow{R} Z_N^* \backslash R(N, k)] - Pr[D(N, k, x) = 1 | x \xleftarrow{R} R(N, k)] \right|.$$

For any PPT D, Pr is negligible.

2.6 k-subgroup Assumption for Π_k

In 1998, Okamoto and Uchiyama proposed P-subgroup assumption to prove their scheme OU98 to be semantic secure [17].

In OU98, $N = P^2Q$, $k = P$. Now we redefine OU98 with λ-acceptable (N, k). We call the new scheme Π_k :

- $Gen(1^\lambda)$:

  (N, k) is λ-acceptable. The length of k is l.

  $Y(N, k) = \left\{ y \middle| P^{a-1} | order_{P^a}(y),\ Q^{b-1} | order_{Q^b}(y) \right\}$, $y \xleftarrow{R} Y(N, k)$.

  $h_0 \xleftarrow{R} Z_N^*$, $h = h_0^k \bmod N$.

  $pk = (N, y, h, l)$, $sk = (P, Q)$.

- $Enc_{pk}(m)$:

  $c = y^m h^x \bmod N$, $m \in \{0, 1\}^{l-1}$, $x \xleftarrow{R} Z_N$.

- $Dec_{sk}(c)$:

  $\langle m \rangle_{P^{k_a}} = \frac{L(c^{P^{a-2}(P-1)} \bmod P^a)}{L(y^{P^{a-2}(P-1)} \bmod P^a)} \bmod P^{k_a}$, $\langle m \rangle_{Q^{k_b}} = \frac{L(c^{Q^{b-2}(Q-1)} \bmod Q^b)}{L(y^{Q^{b-2}(Q-1)} \bmod Q^b)} \bmod Q^{k_b}$.

  Compute $\langle m \rangle_k$ using CRT.

Then we define k-subgroup assumption for Π_k to prove our scheme.

Definition 2.4 (k-subgroup assumption for Π_k)**.**

Given (pk, c), it is hard to decide whether $c \in Enc_{pk}(0)$ or $c \in Enc_{pk}(1)$.

Theorem 2.2. *k-subgroup assumption for $\Pi_k \Leftrightarrow (N,k)$ residuosity assumption.*

Proof.

(1) k-subgroup assumption for $\Pi_k \Rightarrow (N,k)$ residuosity assumption.

D : Input (pk, c). Task: decide whether $c \in Enc_{pk}(0)$ or $c \in Enc_{pk}(1)$.

D' : Input (N, x). Task: decide whether $x \in R(N,k)$ or $x \in Z_N^* \backslash R(N,k)$.

$Adv_{\mathcal{D}'}(1^\lambda) = \left| Pr[D'(N,x) = 1 | x \xleftarrow{R} R(N,k)] - Pr[D'(N,x) = 1 | x \xleftarrow{R} Z_N^* \backslash R(N,k)] \right|$.

D feeds D' with (N, c) and gets the output b' of D'. D outputs b'.

If $c \in Enc_{pk}(0)$, $c \in R(N,k)$.

If $c \in Enc_{pk}(1)$, $c \in Z_N^* \backslash R(N,k)$.

$\text{negl}(\lambda) \geq |Pr[D(pk,c) = 1 | c \xleftarrow{R} Enc_{pk}(0)] - Pr[D(pk,c) = 1 | c \xleftarrow{R} Enc_{pk}(1)]|$

$\geq Adv_{\mathcal{D}'}(1^\lambda)$.

Therefore, k-subgroup assumption for $\Pi_k \Rightarrow (N,k)$ residuosity assumption.

(2) (N,k) residuosity assumption $\Rightarrow$ k-subgroup assumption for Π_k.

D : Input (N, x). Task: decide whether $x \in R(N,k)$ or $x \in Z_N^* \backslash R(N,k)$.

D' : Input (pk, c). Task: decide whether $c \in Enc_{pk}(0)$ or $c \in Enc_{pk}(1)$.

D takes the role of challenger in the challenge game of D'. D randomly choose a bit $b \in \{0,1\}$ and compute a challenge ciphertext $c = y^b x^r \bmod N$, $r \xleftarrow{R} Z_N$.

D feeds D' with (pk, c) and gets the output b' of D'. D outputs b'.

$Adv_{\mathcal{D}'}(1^\lambda) = |Pr[D'(pk,c) = 1 | c \in Enc_{pk}(0)] - Pr[D'(pk,c) = 1 | c \in Enc_{pk}(1)]|$.

If $x \in R(N,k)$, $c \in Enc_{pk}(b)$.

If $x \in Z_N^* \backslash R(N,k)$, $Adv_{\mathcal{D}'}(1^\lambda) = 0$.

$\text{negl}(\lambda) \geq |Pr[D(N,x) = 1 | x \in R(N,k)] - Pr[D(N,x) = 1 | x \in Z_N^* \backslash R(N,k)]|$

$\geq Adv_{\mathcal{D}'}(1^\lambda)$.

Therefore, (N,k) residuosity assumption $\Rightarrow$ k-subgroup assumption for Π_k. $\square$

3 Our Generalized Paillier's Scheme

3.1 KeyGen(1^λ) :

(N,k) is λ-acceptable. The length of k is l.

If $ak_b = bk_a$, $K = k$, otherwise $K = N$.

$Y(N,k) = \left\{ y \middle| P^{a-1} | order_{P^a}(y),\ Q^{b-1} | order_{Q^b}(y) \right\}$.

$P^{a-1} \mid order_{P^a}(y) \Leftrightarrow y^{\varphi(P^a)/P} \not\equiv 1 \bmod P^a$, $Q^{b-1} \mid order_{Q^b}(y) \Leftrightarrow y^{\varphi(Q^b)/Q} \not\equiv 1 \bmod Q^b$.

Randomly choose $y \in Y(N,k)$.

$pk = (N, y, l, K)$, $sk = (P, Q, k)$.

3.2 $\mathrm{Enc}_{pk}(m)$:

If $K = N$, the message space $M = \{0,1\}^{l-1}$, otherwise $M = \{m \,|\, m < K\}$.

Compute the ciphertext $c = y^m x^K \bmod N$, $m \in M$, $x \in Z_N^*$.

3.3 $\mathrm{Dec}_{sk}(c)$:

$k = P^{k_a} Q^{k_b}$, if we can compute $\langle m \rangle_{P^{k_a}}$ and $\langle m \rangle_{Q^{k_b}}$ respectively, then we can compute $\langle m \rangle_k$ using Chinese Remainder Theorem(CRT).

If $k_a \geq 1$, compute $\langle m \rangle_{P^{k_a}}$.

$c = y^m x^K \bmod N$, $c = y^m x^K \bmod P^a$.

$K = k$ or $K = N$, $P^{k_a} | K$.

$c^{\varphi(P^a)/P^{k_a}} \equiv (y^{\varphi(P^a)/P^{k_a}})^m x^{\varphi(P^a)K/P^{k_a}} \equiv (y^{\varphi(P^a)/P^{k_a}})^m \bmod P^a$.

$P^{a-1} | order_{P^a}(y)$, $order_{P^a}(y^{\varphi(P^a)/P^{k_a}}) = \frac{order_{P^a}(y)}{(order_{P^a}(y), \varphi(P^a)/P^{k_a})} = P^{k_a}$.

It is a $(y^{\varphi(P^a)/P^{k_a}}, P^{k_a}, P^a)$ discrete logarithm problem.

Compute $a' = \lfloor \frac{a}{2} \rfloor$.

- If $k_a \leq a'$, $P^{2k_a} | P^a$, then we can compute $\langle m \rangle_{P^{k_a}}$ directly using Theorem 2.1.

- If $k_a > a'$, $\langle m \rangle_{P^{k_a}} = m_1 P^{a'} + m_2$, $0 \leq m_1 < P^{k_a - a'}$, $0 \leq m_2 < P^{a'}$.

  $c^{\varphi(P^a)/P^{a'}} \equiv (y^{\varphi(P^a)/P^{a'}})^m x^{\varphi(P^a)K/P^{a'}} \equiv (y^{\varphi(P^a)/P^{a'}})^{m_2} \bmod P^a$.

  $P^{a-1} | order_{P^a}(y)$, $order_{P^a}(y^{\varphi(P^a)/P^{a'}}) = \frac{order_{P^a}(y)}{(order_{P^a}(y), \varphi(P^a)/P^{a'})} = P^{a'}$.

  It is a $(y^{\varphi(P^a)/P^{a'}}, P^{a'}, P^a)$ discrete logarithm problem. We can compute m_2 using Theorem 2.1.

  $c^{\varphi(P^a)/P^{k_a}} \equiv (y^{\varphi(P^a)/P^{k_a}})^{m_1 P^{a'} + m_2} \bmod P^a$.

$c^{\varphi(P^a)/P^{k_a}}(y^{\varphi(P^a)/P^{k_a}})^{-m_2} \equiv (y^{\varphi(P^a)/P^{k_a-a'}})^{m_1} \bmod P^a$.

$order_{P^a}(y^{\varphi(P^a)/P^{k_a-a'}}) = \frac{order_{P^a}(y)}{(order_{P^a}(y),\varphi(P^a)/P^{k_a-a'})} = P^{k_a-a'}$.

It is a $(y^{\varphi(P^a)/P^{k_a-a'}}, P^{k_a-a'}, P^a)$ discrete logarithm problem. We can compute m_1 using Theorem 2.1.

$\langle m \rangle_{P^{k_a}} = m_1 P^{a'} + m_2$.

If $k_b \geq 1$, compute $\langle m \rangle_{Q^{k_b}}$ in the same way.

Compute $\langle m \rangle_k$ using CRT.

From the decryption algorithm, we can see the number of multiplications for decryption is $O(\lambda)$. It is independent of a and b, so our scheme is faster than other generalized Paillier's schemes in decryption.

4 Security of our Generalized Paillier's Scheme

In this section, we prove that our generalized Paillier's scheme is IND-CPA secure under k-subgroup assumption for Π_k we mentioned in Section 2.6. In fact, if $ak_b = bk_a$, $\frac{k_a}{(k_a,k_b)} \geq a - k_a$, $\frac{k_b}{(k_a,k_b)} \geq b - k_b$, it is IND-CPA secure under (k'^2, k') residuosity assumption, where $k' = P^{\frac{k_a}{(k_a,k_b)}} Q^{\frac{k_b}{(k_a,k_b)}}$. Both Pai99 and DJ01 satisfy this condition. We see DJ01 as a example, $a = b = a$, $k_a = k_b = a-1$, $k' = PQ$, $\frac{k_a}{(k_a,k_b)} \geq a-k_a$, $\frac{k_b}{(k_a,k_b)} \geq b-k_b$. DJ01 is IND-CPA secure under (P^2Q^2, PQ) residuosity assumption through Theorem 4.2. The same result is given in DJ01. In addition to the special case of DJ01, some other cases can use Theorem 4.2 to prove security. For instance, $a = 12$, $b = 8$, $k_a = 9$, $k_b = 6$, $k' = P^3Q^2$, $\frac{k_a}{(k_a,k_b)} \geq a - k_a$, $\frac{k_b}{(k_a,k_b)} \geq b - k_b$, we can prove this scheme is IND-CPA secure under (P^6Q^4, P^3Q^2) residuosity assumption.

Theorem 4.1. *Our generalized Paillier's scheme is IND-CPA secure under k-subgroup assumption for Π_k.*

Proof.

D : Input (N, y, h, l, c) for Π_k. Task: decide whether $c \in Enc_{\Pi,pk}(0)$ or $c \in Enc_{\Pi,pk}(1)$

$\mathcal{A}$ is a PPT IND-CPA adversary for the generalized Paillier's scheme.

D take the role of the challenger of IND-CPA game for the generalized Paillier's scheme. D receives m_0 and m_1 from $\mathcal{A}$ and computes $c_1 = c^{m_0 - m_1} \bmod N$, $c_2 = c_1 y^{m_0} \bmod N$, $c_3 = c_2 x^K \bmod N$, $x \in Z_N^*$.

D feeds $\mathcal{A}$ with c_3 and gets the output b of $\mathcal{A}$. D outputs b.

If $c \in Enc_{\Pi,pk}(0)$, $c_3 \in Enc_{pk}(m_0)$.

If $c \in Enc_{\Pi,pk}(1)$, $c_3 \in Enc_{pk}(m_1)$.

Assume that our generalized Paillier's scheme is not IND-CPA secure: i.e., $\mathcal{A}$ can distinguish $Enc_{pk}(m_0)$ and $Enc_{pk}(m_1)$ with non-negligible probability. Then D can distinguish $Enc_{\Pi,pk}(0)$ and $Enc_{\Pi,pk}(1)$ with non-negligible probability, which is contradictory to k-subgroup assumption for Π_k.

Our generalized Paillier's scheme is IND-CPA secure under k-subgroup assumption for Π_k. □

Theorem 4.2. *For a special case* $ak_b = bk_a$, $\frac{k_a}{(k_a,k_b)} \geq a - k_a$, $\frac{k_b}{(k_a,k_b)} \geq b - k_b$, *compute* $k' = P^{\frac{k_a}{(k_a,k_b)}} Q^{\frac{k_b}{(k_a,k_b)}}$, *our generalized Paillier's scheme is IND-CPA secure under* (k'^2, k') *residuosity assumption.*

Proof.

Theorem 2.2 proves that k-subgroup assumption for $\Pi_k \Leftrightarrow (N, k)$ residuosity assumption.

Theorem 4.1 proves that our generalized Paillier's scheme is IND-CPA secure under k-subgroup assumption for Π_k.

So our generalized Paillier's scheme is IND-CPA secure under (N, k) residuosity assumption.

If we can prove that (k'^2, k') residuosity assumption $\Rightarrow$ (N, k) residuosity assumption, then we can prove this theorem. We prove it in two steps.

(1) (k'^2, k') residuosity assumption $\Rightarrow$ (k'^{s+1}, k'^s) residuosity assumption

We prove it by mathematical induction.

When $s = 1$, it is obvious. Suppose (k'^2, k') residuosity assumption $\Rightarrow$ (k'^{t+1}, k'^t) residuosity assumption for $t \geq 2$. We just need to prove that (k'^{t+1}, k'^t) residuosity assumption $\Rightarrow$ (k'^{t+2}, k'^{t+1}) residuosity assumption.

D: Input (k', t, x). Task: decide whether $x \in R(k'^{t+1}, k'^{t})$.

D': Input (k', t, x). Task: decide whether $x \in R(k'^{t+2}, k'^{t+1})$.

D computes $x' = x^{k'} \bmod k'^{t+2}$ and feeds D' with (k', t, x').

If $x \in R(k'^{t+1}, k'^{t})$, $x = ck'^{t+1} + r^{k'^{t}}, x' \equiv r^{k'^{t+1}} \bmod k'^{t+2}$, $x' \in R(k'^{t+2}, k'^{t+1})$.

If D' can distinguish $R(k'^{t+2}, k'^{t+1})$ and $Z^*_{k'^{t+2}} \backslash R(k'^{t+2}, k'^{t+1})$ with non-negligible probability, then D can distinguish $R(k'^{t+1}, k'^{t})$ and $Z^*_{R(k'^{t+1}, k'^{t})} \backslash R(k'^{t+1}, k'^{t})$ with non-negligible probability, which is contradictory to the assumption. So (k'^{t+1}, k'^{t}) residuosity assumption $\Rightarrow$ (k'^{t+2}, k'^{t+1}) residuosity assumption.

(2) $(k'^{(k_a,k_b)+1}, k'^{(k_a,k_b)})$ residuosity assumption $\Rightarrow$ (N, k) residuosity assumption

$k'^{(k_a,k_b)+1} = P^{k_a} P^{\frac{k_a}{(k_a,k_b)}} Q^{k_b} P^{\frac{k_b}{(k_a,k_b)}}$, $k'^{(k_a,k_b)} = k$.

$\frac{k_a}{(k_a,k_b)} \geq a - k_a$, $\frac{k_b}{(k_a,k_b)} \geq b - k_b$, so $N | k'^{(k_a,k_b)+1}$.

It is easy to prove (dN, k) residuosity assumption $\Rightarrow$ (N, k) residuosity assumption, so we prove that $(k'^{(k_a,k_b)+1}, k'^{(k_a,k_b)})$ residuosity assumption $\Rightarrow$ (N, k) residuosity assumption. □

5 Conclusions

In this paper, we propose a generalization of Paillier's scheme and all the Paillier's schemes to our knowledge are special cases of our scheme. In our scheme, $N = P^a Q^b$, $k = P^{k_a} Q^{k_b}$, so the ciphertext expansion $r = \frac{a+b}{k_a+k_b}$. We use a general y satisfying $P^{a-1} | order_{P^a}(y)$, $Q^{b-1} | order_{Q^b}(y)$ instead of some special y such as $y = 1 + PQ \bmod N$. We also propose a very simple decryption algorithm which is more efficient than other generalization algorithms(see Table 2). Our generalized Paillier's scheme is IND-CPA secure under k-subgroup assumption for Π_k which is equivalent to (N, k) Residuosity assumption. Other generalization schemes are special cases of our scheme which we can prove the security under (k'^2, k') residuosity assumption where $k' = P^{\frac{k_a}{(k_a,k_b)}} Q^{\frac{k_b}{(k_a,k_b)}}$. So our scheme is as secure as other generalization schemes.

Table2: Comparison of efficiency with equal security parameter λ

Scheme	**DJ01**	**OAS08**	**Our General Scheme**
number of multiplications for decryption	$O(a\lambda + a^2)$	$O(a + \lambda)$	$O(\lambda)$

References

[1] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. *Advances in Cryptology Eurocrypt*, 547(1):223–238, 1999.

[2] Dario Catalano, Rosario Gennaro, Nick Howgrave-Graham, and Paul Nguyen. Pailliers cryptosystem revisited. *Proc. of ACM CCS*, pages 206–214, 2001.

[3] MadsJohan Jurik. Extensions to the paillier cryptosystem with applications to cryptological protocols. 2003.

[4] Nguyen Lan, Reihaneh Safavi-Naini, and Kaoru Kurosawa. Verifiable shuffles: A formal model and a paillier-based efficient construction with provable security. In *Applied Cryptography and Network Security, Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, Proceedings*, 2004.

[5] Takao Onodera and Keisuke Tanaka. Shufle for paillier's encryption scheme. *Ieice Transactions on Fundamentals of Electronics Communications Computer Sciences*, E88-A(5):1241–1248, 2005.

[6] Katja Schmidt-Samoa and Tsuyoshi Takagi. Paillier's cryptosystem modulo p2q and its applications to trapdoor commitment schemes. In *Proceedings*

of the 1st international conference on Progress in Cryptology in Malaysia, 2005.

[7] Padmavathamma MP.Vasudeva Reddy. A generalization and simplification of a variant of a pailliers public key cryptosystem with applications to electronic voting. 2008.

[8] P.Y.A. Ryan. Prêt à voter with paillier encryption. *Mathematical Computer Modelling*, 48(9-10):1646–1662, 2008.

[9] Ivan DamgåRd, Mads Jurik, and JesperBuus Nielsen. A generalization of paillier's public-key system with applications to electronic voting. 9(6):371–385, 2010.

[10] PeiYih Ting and ChiaHuei Hseu. A secure threshold paillier proxy signature scheme. *Journal of Zhejiang University Science C*, (03):58–65, 2010.

[11] ZhiWei Chen, D.U. Min, YaTao Yang, and L.I. Zi-Chen. Homomorphic cloud computing scheme based on rsa and paillier. *Computer Engineering*, 2013.

[12] Dario Catalano, Antonio Marcedone, and Orazio Puglisi. Authenticating computation on groups: New homomorphic primitives and applications. In *International Conference on the Theory and Application of Cryptology and Information Security*, 2014.

[13] Zhengjun Cao and Lihua Liu. The paillier's cryptosystem and some variants revisited. *Computer Science*, 2015.

[14] Carmit Hazay, GertLsse Mikkelsen, Tal Rabin, Tomas Toft, and AngeloAgatino Nicolosi. Efficient rsa key generation and threshold paillier in the two-party setting. *Journal of Cryptology*, 32(2):265–323, 2019.

[15] Ivan Damgård and Mads Jurik. A generalisation, a simplication and some applications of paillier's probabilistic public-key system. In *Public Key Cryptography*, pages 119–136, 2001.

[16] O.O. Obi, F.H. Ali, and E.Stipidis. Explicit expression for decryption in a generalisation of the paillier scheme. *Iet Information Security*, 1(4):163–166, 2008.

[17] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 308–318, 1998.