The Case of Small Prime Numbers Versus the Okamoto-Uchiyama Cryptosystem

George Teşeleanu[1,2]

[1] Advanced Technologies Institute
10 Dinu Vintilă, Bucharest, Romania
tgeorge@dcti.ro
[2] Simion Stoilow Institute of Mathematics of the Romanian Academy
21 Calea Grivitei, Bucharest, Romania

Abstract. In this paper we study the effect of using small prime numbers within the Okamoto-Uchiyama public key encryption scheme. We introduce two novel versions and prove their security. Then we show how to choose the system's parameters such that the security results hold. Moreover, we provide a practical comparison between the cryptographic algorithms we introduced and the original Okamoto-Uchiyama cryptosystem.

Keywords: public key encryption, p-subgroup, provable security

1 Introduction

The Okamoto-Uchiyama cryptosystem was introduced in [19] and the authors proved that inverting the encryption function is equivalent to factoring. Unlike most factoring based schemes, the Okamoto-Uchiyama encryption is based on factoring numbers of the form p^2q instead of pq, where p and q are prime numbers. Another important security results is that the scheme is semantically secure if the p-subgroup assumption holds. This assumption is comparable to the quadratic residue and high degree residue assumptions [19]. We underline that the Okamoto-Uchiyama cryptosystem is partially homomorphic with respect to message addition and supports ciphertext randomization.

Shortly after the publication of [19], Coron, Naccache and Paillier introduced in [9] a variant of the Okamoto-Uchiyama cryptosystem that reduces the complexity of decryption. The authors claim, without proof, that the scheme retains the same security as the original scheme. We revisit their claims and we argue that is not obvious why the scheme is as secure as factoring. Therefore, in our opinion, the equivalence of Coron *et al.*'s variant to factoring remains an open problem. We also show that the semantic security of this variant is linked to a special kind of p-subgroup problem.

Another variant was introduced in [8], which also aims at reducing decryption complexity. They achieve this by choosing a special type of generator, and therefore instead of doing two exponentiations and a modular inversion, Choi, Choi and Won manage to decrypt using only an exponentiation. Similar to Coron *et* *al.*, Choi *et al.* claim that their variant is equivalent to factoring. Their claim is refuted in [23] and it remains an open problem to prove that Choi *et al.*'s encryption scheme is not invertible. Note that the Okamoto-Uchiyama cryptosystem allows us to precompute an exponentiation and the corresponding inverse, and thus reduce the complexity of decryption to only one exponentiation. Therefore, we can obtain a decryption time similar to Choi *et al.* at the cost of memorizing an element modulo n and without sacrificing equivalence to factoring.

Since one of the most important features of the Okamoto-Uchiyama cryptosystem is its equivalence to factoring, we aim at finding a way to decrease decryption times, while retaining the cryptosystem's one-way equivalence to some form of factoring. Therefore, we introduce two variants of the scheme: an unbalanced version and a multiprime one. In the first version we show how to decrease the size of p while keeping the system secure and implicitly decreasing the complexity of decryption. The only cryptosystem related to this version is called the unbalanced RSA [24]. Just like our version, the scope is to reduce p, while keeping the system secure. Compared to the unbalanced RSA, the one-way property of the unbalanced Okamoto-Uchiyama is equivalent to factoring.

In the multiprime version, we increase the number of factors while keeping the size of the modulus constant and we manage to prove that inverting encryption is equivalent to computing the square-free factor of n. In the literature, we can find two related cryptosystems: the multiprime RSA [4,22] and the multiprime Joye-Libert [16,26]. The philosophy behind the multiprime RSA is the same as ours, while in the case of the multiprime Joye-Libert, the authors use multiple primes, but they increase the size of the modulus. Note that none of these systems is equivalent to factoring, while our variant is equivalent to partially factoring the modulus. According to [15], the multiprime RSA was implemented in Wireless Transport Layer Security protocol in order to decrease decryption times by use of parallelism. The same trick can be used to speed up the multiprime Okamoto-Uchiyama cryptosystem.

In the final section of our paper, we analyze the complexity of the two novel variants. Then we compare decryption times for all versions of the Okamoto-Uchiyama scheme that are equivalent to some form of factoring. If parallelization is possible, then the multiprime variant is to be preferred since it has a larger message space. Otherwise, for messages under a certain threshold, the unbalanced version has faster decryption times and once the threshold is crossed, the multiprime variant surpasses the efficiency of the unbalanced one.

For completeness we also provide an unbalanced and a multiprime version of the Coron-Naccache-Paillier cryptosystem. Then we prove their semantic security and finally we analyse their performance. We do not present a similar treatment for the Choi-Choi-Won cryptosystem, since an optimization equivalent with theirs can be achieved by simpler means, as states above.

Structure of the paper. In Section 2 we introduce notations, definitions and lemmas used throughout the paper. The original Okamoto-Uchiyama scheme is described in Section 3. In Sections 4 and 5 we present two novel versions of the Okamoto-Uchiyama scheme. The Coron-Naccache-Paillier variant is tackled in

Section 6. A performance analysis of some of the Okamoto-Uchiyama variants is provided in Section 7. We conclude in Section 8. A multiprime version of the Coron-Naccache-Paillier variant is given in Appendix A.

2 Preliminaries

Notations. Throughout the paper, λ denotes a security parameter. By $|n|$ we denote the size of n in bits. We use the notation $x \overset{\$}{\leftarrow} X$ when selecting a random element x from a sample space X. We denote by $x \leftarrow y$ the assignment of the value y to the variable x. The probability that event E happens is denoted by $Pr[E]$. Probabilistic polynomial-time algorithms are referred to as PPT algorithms. The set of integers $\{0, \ldots, a-1\}$ is further denoted by $[0, a)$. For shorthand, we denote the set $[0, a+1)$ by $[0, a]$. Multidimensional vectors $v = (v_0, \ldots, v_{s-1})$ are represented as $v = \{v_i\}_{i \in [0,s)}$.

2.1 Computational Complexity

In order to determine the computational complexity of our proposed schemes, we use the complexities of the mathematical operations listed in Table 1. These complexities are in accordance with the algorithms presented in [10, 17]. We do not use the explicit complexity of multiplication, but instead we refer to it as $M(\cdot)$ for clarity. When presenting the complexity of performing an exponentiation we assume that the exponent has k bits. Also, in the case of the Chinese remainder theorem (CRT) we consider that the resulting modulus has μ bits and that we have r moduli.

Operation	Complexity
Multiplication	$M(\mu) = \mathcal{O}(\mu \log \mu \log \log \mu)$
Exponentiation	$\mathcal{O}(kM(\mu))$
Modular inverse	$\mathcal{O}(\mu M(\mu))$
CRT	$\mathcal{O}(\log r M(\mu))$

Table 1: Computational complexity for μ-bit numbers

2.2 Number Theoretic Prerequisites

We further present some definitions and lemmas from [19] that are needed for describing the public key encryption schemes presented in this paper.

Definition 1. *Let p be an odd prime. We define the p-Sylow subgroup of $\mathbb{Z}_{p^2}^*$ as*

$$\Gamma = \{x \in \mathbb{Z}_{p^2}^* \mid x \equiv 1 \bmod p\}.$$

Remark that $\mathbb{Z}_{p^2}^*$ is a cyclic group of order $p(p-1)$. Therefore, we obtain the following consequence.

Lemma 1. *Let p be an odd prime. Then the p-Sylow subgroup of $\mathbb{Z}_{p^2}^*$ has cardinality p.*

Lemma 2. *Let p be an odd prime, $g \in \mathbb{Z}_{p^2}^*$ and Γ the p-Sylow subgroup of $\mathbb{Z}_{p^2}^*$. If $g^{p-1} \bmod p^2$ has order p, then $g^{p-1} \in \Gamma$.*

Definition 2. *Let p be an odd prime and Γ the p-Sylow subgroup of $\mathbb{Z}_{p^2}^*$. We define the logarithmic function $L(\cdot)$ on Γ as*

$$L(x) = \frac{x-1}{p}, \text{ for any } x \in \Gamma.$$

Lemma 3. *Let $L(\cdot)$ be the logarithmic function on Γ. Then for any $x, y \in \Gamma$ we have $L(xy) \equiv L(x) + L(y) \bmod p$.*

Lemma 4. *Let $L(\cdot)$ be the logarithmic function on Γ. Also, let $x \in \Gamma$ such that $L(x) \neq 0$ and $y \equiv x^m \bmod p^2$, where $m \in \mathbb{Z}_p$. Then the following holds*

$$m \equiv \frac{L(y)}{L(x)} \equiv \frac{y-1}{x-1} \bmod p.$$

2.3 Public Key Encryption

A *public key encryption* (PKE) scheme usually consists of three PPT algorithms: *Setup*, *Encrypt* and *Decrypt*. The *Setup* algorithm takes as input a security parameter and outputs the public key as well as the matching secret key. *Encrypt* takes as input the public key and a message and outputs the corresponding ciphertext. The *Decrypt* algorithm takes as input the secret key and a ciphertext and outputs either a valid message or an invalidity symbol (if the decryption failed).

Definition 3 (Indistinguishability under Chosen Plaintext Attacks - IND-CPA). *The security model against chosen plaintext attacks for a PKE scheme is captured in the following game:*

Setup(λ): The challenger C generates the public key, sends it to adversary A and keeps the matching secret key to himself.
Query: Adversary A sends to C two equal length messages m_0, m_1. The challenger flips a coin $b \in \{0,1\}$ and encrypts m_b. The resulting ciphertext c is sent to the adversary.
Guess: In this phase, the adversary outputs a guess $b' \in \{0,1\}$. He wins the game if $b' = b$.

The advantage of an adversary A attacking a PKE scheme is defined as

$$ADV_A^{\textit{IND-CPA}}(\lambda) = |Pr[b = b'] - 1/2|$$

where the probability is computed over the random bits used by C and A. A PKE scheme is IND-CPA *secure, if for any PPT adversary A the advantage $ADV_A^{\textit{IND-CPA}}(\lambda)$ is negligible.*

3 The Okamoto-Uchiyama PKE scheme

The Okamoto-Uchiyama scheme was introduced in [19] and the authors prove that inverting the encryption function is as hard as factoring. The scheme was also proven IND-CPA secure in the standard model under the p-subgroup assumption[3]. We shortly describe the algorithms of the Okamoto-Uchiyama cryptosystem.

Setup(λ): Generate two distinct large prime numbers p, q such that $|p| = |q| = \lambda$ and compute $n = p^2q$. Randomly select $g \in \mathbb{Z}_n$ such that $g_p \equiv g^{p-1} \bmod p^2$ has order p in $\mathbb{Z}^*_{p^2}$. Let $h \equiv g^n \bmod n$. Output the public key $pk = (n, g, h, \lambda)$ and the corresponding secret key $sk = (p, q)$.

Encrypt(pk, m): To encrypt a message $m \in [0, 2^{\lambda-1})$ we choose $r \overset{\$}{\leftarrow} \mathbb{Z}_n$ and compute $c \equiv g^m h^r \bmod n$. Output the ciphertext c.

Decrypt(sk, c): Compute $c_p \equiv c^{p-1} \bmod p^2$, $g_p \equiv g^{p-1} \bmod p^2$ and recover m from the relation

$$m \equiv \frac{L(c_p)}{L(g_p)} \bmod p.$$

4 The Unbalanced Okamoto-Uchiyama PKE scheme

In the unbalanced Okamoto-Uchiyama scheme we reduce the size of p (denoted λ_p), while keeping the size of n constant (denoted λ_n). This modification only impacts the description of the *Setup* and *Encrypt* algorithms, which we briefly describe below. Therefore, we have $\lambda_n = 2\lambda_p + \lambda_q$, where $\lambda_q = |q|$ and $\lambda_p \leq \lambda_q$. Note that when $\lambda_p = \lambda_q$ we obtain the Okamoto-Uchiyama cryptosystem, which we further refer to as the balanced Okamoto-Uchiyama scheme.

Setup(λ_p, λ_q): Generate two distinct large prime numbers p, q such that $|p| = \lambda_p$ and $|q| = \lambda_q$. Let $n = p^2q$. Randomly select $g \in \mathbb{Z}_n$ such that $g_p \equiv g^{p-1} \bmod p^2$ has order p in $\mathbb{Z}^*_{p^2}$. Let $h \equiv g^n \bmod n$. Output the public key $pk = (n, g, h, \lambda_p)$ and the corresponding secret key $sk = (p, q)$.

Encrypt(pk, m): To encrypt a message $m \in [0, 2^{\lambda_p-1})$ we choose $r \overset{\$}{\leftarrow} \mathbb{Z}_n$ and compute $c \equiv g^m h^r \bmod n$. Output the ciphertext c.

Remark 1. Modifying the size of p does not impact the security proofs from [19]. Therefore, as long as factoring is hard, the unbalanced version is secure. We discuss how to choose λ_p such that factoring remains difficult in Section 7.

[3] To the author's knowledge, the only method for breaking this assumption is to know the factorisation of n.

5 The Multiprime Okamoto-Uchiyama PKE scheme

5.1 Description

We further describe the multiprime Okamoto-Uchiyama encryption scheme. In this case we split up n into multiple primes. Therefore, we have $\lambda_n = 2t\lambda_p + \lambda_q$, where λ_q is the size of the square free prime factor q and $\lambda_p \leq \lambda_q$. Note that the case $t = 1$ and $\lambda_p = \lambda_q$ corresponds to the original cryptosystem [19]. Also, if we set $t = 1$, we obtain the unbalanced version.

Setup(λ_p, λ_q): Generate $t+1$ distinct large prime numbers $p_1, \dots, p_t, q$ such that $|p_1| = \ldots = |p_t| = \lambda_p$ and $|q| = \lambda_q$. Let $n = p_1^2 \ldots p_t^2 q$. Randomly select $g \in \mathbb{Z}_n$ such that for any $i \in [1,t]$ the element $g_{p_i} \equiv g^{p_i - 1} \bmod p_i^2$ has order p_i in $\mathbb{Z}^*_{p_i^2}$. Let $h \equiv g^n \bmod n$. Output the public key $pk = (n, g, h, \lambda_p, t)$ and the corresponding secret key $sk = (P, q)$, where $P = \{p_i\}_{i \in [1,t]}$.

Encrypt(pk, m): To encrypt a message $m \in [0, 2^{\lambda_p t - 1})$, we choose $r \xleftarrow{\$} \mathbb{Z}_n$ and compute $c \equiv g^m h^r \bmod n$. Output the ciphertext c.

Decrypt(sk, c): For each $i \in [1,t]$ compute $c_{p_i} \equiv c^{p_i - 1} \bmod p_i^2$, $g_{p_i} \equiv g^{p_i - 1} \bmod p_i^2$ and recover m_i from the relation

$$m_i \equiv \frac{L(c_{p_i})}{L(g_{p_i})} \bmod p_i.$$

Let $n' = p_1 \ldots p_t$. Using the CRT, compute the unique $m \in \mathbb{Z}_{n'}$ such that $m \equiv m_i \bmod p_i$.

Correctness. The recovery of m is possible due to the following relation

$$c_{p_i} \equiv c^{p_i - 1} \equiv (g^m h^r)^{p_i - 1} \equiv g^{m(p_i-1)} (g^{rn/p_i})^{p_i(p_i-1)} \equiv g^{m(p_i-1)} \bmod p_i^2,$$

which according to Lemmas 2 and 4 implies

$$m \equiv \frac{L(c_{p_i})}{L(g_{p_i})} \bmod p_i.$$

Optimizations. In the *Setup* phase, we have to compute a special type of g. An efficient way to perform this step is to first randomly select $g_i \xleftarrow{\$} \mathbb{Z}^*_{p_i^2}$ such that $g_i^{p_i - 1} \bmod p_i^2$ has order p_i for $i \in [1,t]$ and then choose a random element $g_{t+1} \xleftarrow{\$} \mathbb{Z}^*_q$. Afterwards use the CRT to compute an element $g \in \mathbb{Z}^*_n$ such that $g \equiv g_i \bmod p_i^2$ for all i and that $g \equiv g_{t+1} \bmod q$.

Another possible optimisation is to memorize the values used during the *Decrypt* algorithm that are known beforehand. More precisely, during the *Setup* phase we can precompute the values $L(g_{p_i})^{-1} \bmod p_i$, p_i^2 and n', where $i \in [1,t]$. These values can enhance the secret key, and therefore can be used to speed up the decryption process at the cost of using more memory.

5.2 Security Analysis

In this section we first introduce a novel security assumption called the square-free factor problem and prove that inverting the encryption function of our proposal is as hard as breaking this assumption. Note that, when $t = 1$, the SFF assumption is equivalent with factoring the modulus. Then we generalise the p-subgroup problem stated in [19] and prove the IND-CPA security of our proposal. Note that for simplicity we assume, without losing generality, that $\lambda_p = \lambda_q = \lambda$ when conducting the security analysis.

Definition 4 (Square-Free Factor - SFF). *Choose $t+1$ distinct large prime numbers $p_1, \ldots, p_t, q \geq 2^\lambda$. Let A be a PPT algorithm that returns an integer. We define the advantage*

$$ADV_A^{\text{SFF}}(\lambda) = Pr[A(n) = q \mid n = p_1^2 \ldots p_t^2 q].$$

The Square-Free Factor assumption states that for any PPT algorithm A the advantage $ADV_A^{\text{SFF}}(\lambda)$ is negligible.

Theorem 1. *Inverting the encryption algorithm of the multiprime Okamoto-Uchiyama PKE is intractable if the SFF assumption is intractable.*

Proof. Let's assume that there exists a PPT adversary A that given a ciphertext c recovers m with non-negligible probability. We further construct another PPT algorithm B that given as input n can find the prime factor q with non-negligible probability.

The first step of B is to generate the elements g and h that are needed to run A. Therefore, B randomly chooses $g \xleftarrow{\$} \mathbb{Z}_n^*$ and computes $h \equiv g^n \bmod n$. Note that $g^{p_i-1} \bmod p_i^2$ has order p_i with probability $(p_i-1)/p_i$. Therefore, g and h have the same distribution as in the original scheme with an overwhelming probability $(p_1-1)\ldots(p_t-1)/(p_1\ldots p_t)$.

In the next step B constructs a "ciphertext" c'. More precisely, B randomly generates $z' \xleftarrow{\$} \mathbb{Z}_n$ and computes $c' \equiv g^{z'} \bmod n$. We further show the relation between the distribution of "ciphertexts" c' and the distribution of real ciphertexts c.

Let $c \equiv g^{m+nr} \bmod n$ be a real ciphertext. We denote by $z = m + nr$. Let the order of g modulo $p_1, \ldots, p_t$ be $p_1p_1', \ldots, p_tp_t'$, respectively, while the order modulo q is denoted as q'. Let $\ell = \operatorname{lcm}(p_1', \ldots, p_t', q')$.

Then the distribution of c is given by the distribution of $z = (z_1, z_2)$, where we have $z_1 \equiv z \bmod p_1 \ldots p_t$ and $z_2 \equiv z \bmod \ell$. Similarly, we define $z' = (z_1', z_2')$ as $z_1' \equiv z' \bmod p_1 \ldots p_t$ and $z_2' \equiv z' \bmod \ell$. Remark that z_1 is randomly distributed in $[0, 2^{\lambda t - 1})$, while z_1' is randomly distributed in $\mathbb{Z}_{p_1 \ldots p_t}$, which is roughly the size of $[0, 2^{\lambda t})$. Therefore, the probability of obtaining a given z_1 is at most twice than that of z_1'. Hence, a non-negligible fraction of z_1 is also non-negligible in z_1'.

Note that $\gcd(n, \ell) = 1$, since $\gcd(p_i, \ell) = 1$ and $\gcd(q, \ell) = 1$. Therefore, when z_1 and z_1' are fixed, we have that the distributions of z_2 and z_2' are statistically close.

Given the values (g, h, c') generated by B, adversary A will output a message m with non-negligible probability. Bear in mind that m should satisfy $m < 2^{\lambda t-1}$ and $z' \equiv m \bmod p_1 \dots p_t$. The probability of z' to be greater or equal than $2^{\lambda t-1}$ is

$$\frac{n - 2^{\lambda t-1}}{n} \simeq \frac{2^{\lambda(2t+1)} - 2^{\lambda t-1}}{2^{\lambda(2t+1)}} = \frac{2^{\lambda t-1}(2^{\lambda t+\lambda+1} - 1)}{2^{\lambda t-1}2^{\lambda t+\lambda+1}} \simeq 1,$$

which is non-negligible. Therefore, with overwhelming probability we have $z' \not\equiv m \bmod n$, and thus $\gcd(z' - m, n) \neq 1$.

We claim that $\gcd(z' - m, n) = p_1 \dots p_t$ in almost all the cases. Let $i \in [1, t]$. The property $z' - m \equiv 0 \bmod p_i$ is equivalent with $z' - m = \alpha p_i$, where $\alpha > 0$. If $\alpha p_i \equiv 0 \bmod p_i^2$ then we have that $\alpha \equiv 0 \bmod p_i$. Therefore, the probability of having $z' - m \not\equiv 0 \bmod p_i^2$ is $(p_i - 1)/p_i$. In the case of q we have $z' - m \not\equiv 0 \bmod q$ with probability $(q-1)/q$. Hence, we obtain that $\gcd(z' - m, n) = p_1 \dots p_t$ with overwhelming probability $(p_1 - 1) \dots (p_t - 1)(q - 1)/(p_1 \dots p_t q)$. Therefore, B computes $q = n/(p_1 \dots p_t)^2$ with non-negligible probability. $\square$

Definition 5 (*p*-Subgroups - PS). *Choose $t + 1$ distinct large prime numbers $p_1, \dots, p_t, q \geq 2^\lambda$ and compute $n = p_1^2 \dots p_t^2 q$. Let Γ_i be the p_i-Sylow subgroup of $\mathbb{Z}^*_{p_i^2}$. We define the set $\Delta_i = \{x \in \mathbb{Z}^*_n \mid x^{p_i-1} \in \Gamma_i\}$. We denote by $\Delta = \Delta_1 \cap \dots \cap \Delta_t$ and $\bar{\Delta} = \mathbb{Z}^*_n \setminus \Delta$. Let A be a PPT algorithm that returns 1 on input (x, n) if $x \in \Delta$. We define the advantage*

$$ADV^{PS}_A(\lambda) = \left| Pr[A(x,n) = 1 | x \xleftarrow{\$} \Delta] - Pr[A(x,n) = 1 | x \xleftarrow{\$} \bar{\Delta}] \right|.$$

The p-Subgroups assumption states that for any PPT algorithm A, the advantage $ADV^{PS}_A(\lambda)$ is negligible.

Theorem 2. *The multiprime Okamoto-Uchiyama PKE is* IND-CPA *secure if and only if the* PS *assumption is intractable.*

Proof. We further denote by $E(m)$ the encryption of a message m. Note that breaking the PS assumption is equivalent with distinguishing between $E(0)$ and $E(1)$. To see that we first compute

$$E(0)^{p_i-1} \equiv (g^0 h^{r_0})^{p_i-1} \equiv g^{r_0 n(p_i-1)} \equiv 1 \bmod p_i^2,$$

and

$$E(1)^{p_i-1} \equiv (g^1 h^{r_1})^{p_i-1} \equiv g^{p_i-1} g^{r_1 n(p_i-1)} \equiv g^{p_i-1} \bmod p_i^2.$$

This implies that the order of $E(0)^{p_i-1} \bmod p_i^2$ divides $p_i - 1$ and the order of $E(1)^{p_i-1} \bmod p_i^2$ is p_i since $g^{p_i-1} \bmod p_i^2$ has order p_i. Therefore, $E(0)^{p_i-1} \in \bar{\Delta}$ and $E(1)^{p_i-1} \in \Delta$, for all $i \in [1, t]$.

Now, let's assume that there exists a PPT adversary A that can distinguish between $E(0)$ and $E(1)$. We further construct another PPT algorithm B that can break the IND-CPA security of our proposal.

Once B receives a ciphertext c, he computes the value $x \equiv cg^{-m_0} \bmod n$ and chooses $r \xleftarrow{\$} \mathbb{Z}_n$. Let $\bar{g} \equiv g^{(m_1-m_0)+rn} \bmod n$, $\bar{h} \equiv \bar{g}^n \bmod n$ and $\ell = \operatorname{lcm}(p_1-1,\dots,p_t-1,q-1)$. We denote by $\bar{E}(m)$ the encryption of m using $\bar{g}$ and $\bar{h}$. If B receives the encryption of m_0, we have

$$x \equiv h^{r_0} \equiv \bar{h}^{r_0/(m_1-m_0+rn)} \bmod n,$$

and

$$x \equiv g^{m_1-m_0}h^{r_1} \equiv \bar{g}\bar{h}^{(r_1-r)/(m_1-m_0+rn)} \bmod n,$$

otherwise. Note that since $\gcd(n,\ell)=1$, m_1-m_0+rn is randomly distributed in $\mathbb{Z}_\ell$. Also, m_1-m_0+rn is invertible modulo ℓ with non-negligible probability. Therefore, x is either $\bar{E}(0)$ or $\bar{E}(1)$ with overwhelming probability.

We further prove that with a non-negligible probability the order of $\bar{g}_{p_i} \equiv \bar{g}^{p_i-1} \bmod p_i^2$ is p_i for all $i \in [1,t]$. Remark that $\gcd(m_1-m_0,p_1\dots p_t)=1$ with probability $(p_1-1)\dots(p_t-1)/(p_1\dots p_t)$. Now we assume that there exist $k_i < p_i$ such that $\bar{g}_{p_i}^{k_i} \equiv 1 \bmod p_i$. Then $k_i(m_1-m_0) \equiv 0 \bmod p_i$ for all $i \in [1,t]$. This implies that $k_1\dots k_t(m_1-m_0) \equiv 0 \bmod p_1\dots p_t$. Since m_1-m_0 is coprime with the modulus, we obtain that $k_1\dots k_t \equiv 0 \bmod p_1\dots p_t$, which leads to a contradiction. Therefore, with an overwhelming probability we have $\bar{g} \in \Delta$.

On input $(\bar{g},\bar{h},x)$, adversary A outputs the correct bit b with non-negligible probability. Algorithm B simply relays b and according to the arguments presented above, it guesses correctly whether c is the encryption of m_0 or m_1.

We further prove the converse statement. Thus, let's assume that there exists a PPT adversary A that guesses correctly if a ciphertext encrypts either m_0 or m_1. We assume, without loss of generality, that $m_0 < m_1$. We will construct a PPT machine B which can distinguish between $E(0)$ and $E(1)$.

Given a ciphertext c algorithm B computes $x \equiv c^{m_1-m_0}g^{m_0+nr} \bmod n$, where $r \xleftarrow{\$} \mathbb{Z}_n$. If B receives an encryption of 0, we have

$$x \equiv (h^{r_0})^{m_1-m_0}g^{m_0+nr} \equiv g^{m_0}h^{r_0(m_1-m_0)+r} \bmod n$$

and

$$x \equiv (gh^{r_1})^{m_1-m_0}g^{m_0+nr} \equiv g^{m_1}h^{r_1(m_1-m_0)+r} \bmod n.$$

Therefore, x is either $E(m_0)$ or $E(m_1)$.

On input (g,h,x) adversary A outputs a bit b. Therefore, if algorithm B outputs b, then with non-negligible probability it guesses correctly whether c is the encryption of 0 or 1. $\square$

6 The Coron-Naccache-Paillier PKE scheme

To decrease the complexity of the decryption process, the authors of [9] introduce a slightly modified version of the Okamoto-Uchiyama encryption scheme. We further describe the Coron-Naccache-Paillier optimisation.

Setup(λ, λ_u): Generate a large prime number u such that $|u| = \lambda_u$. Also, generate two distinct large prime numbers p, q such that $|p| = |q| = \lambda$ and $p-1 = uv$. Let $n = p^2 q$. Randomly select $g \in \mathbb{Z}_n$ such that $g_p \equiv g^{p-1} \bmod p^2$ has order p in $\mathbb{Z}^*_{p^2}$. Compute $G \equiv g^v \in \mathbb{Z}^*_n$ and $h \equiv G^n \bmod n$. Output the public key $pk = (n, G, H, \lambda)$ and the corresponding secret key $sk = (p, q)$.

Encrypt(pk, m): To encrypt a message $m \in [0, 2^{\lambda_u - 1})$, we choose $r \overset{\$}{\leftarrow} \mathbb{Z}_n$ and compute $c \equiv G^m H^r \bmod n$. Output the ciphertext c.

Decrypt(sk, c): Compute $c_p \equiv c^u \bmod p^2$, $g_p \equiv G^u \bmod p^2$ and recover m from the relation

$$m \equiv \frac{L(c_p)}{L(g_p)} \bmod p.$$

Remark 2. In the original paper [9], the authors encrypt messages of size $\lambda - 1$, but that leads to an incorrect decryption since the order of G is $u < 2^{\lambda-1}$, and thus a wrap-around of the message is possible. This problem is fixed in our description.

Remark 3. In [9], the authors claim without proof that their optimisation is equivalent with factoring n. They only state that "equivalence to factoring is easily derived from the original security proof included in [19]". When we tried to follow the same line of reasoning as in Theorem 1, the following problem occurred. When choosing G randomly[4] from $\mathbb{Z}^*_n$, the probability of $G^u \bmod p^2$ having order p is

$$\frac{u}{p} \simeq \frac{2^{\lambda_u}}{2^{\lambda}} = \frac{2^{\lambda_u}}{2^{\lambda - \lambda_u}},$$

which is negligible[5]. Therefore, the proof breaks down because G^u will almost never have the correct order, namely p, leading to the simulation failing to generate the correct distributed G with non-negligible probability. Hence, we consider that the optimisation is not equivalent to factoring, until proven otherwise.

Remark 4. In the original paper [9], the IND-CPA security of the scheme is claimed without proof to be equivalent with the PS assumption as introduced in [19]. In reality, the claim is partially true. More precisely, the IND-CPA security can be linked to the following version of the PS assumption[6].

[4] In [19, Theorem 6], this step is equivalent to choosing $g \overset{\$}{\leftarrow} \mathbb{Z}^*_n$.
[5] Compared with the non-negligible value of $(p-1)/p$ as in [19, Theorem 6].
[6] see Appendix A for the proof

Definition 6 (Strong Prime p-Subgroups - SP-PS). *Choose $t+1$ distinct large prime numbers $p_1, \ldots, p_t, q \geq 2^\lambda$ such that $p_i - 1$ has a large prime factor denoted u_i for all $i \in [1,t]$. Let Γ_i be the p_i-Sylow subgroup of $\mathbb{Z}^*_{p_i^2}$. We define the set $\Omega_i = \{x \in \mathbb{Z}_n \mid x^{u_i} \in \Gamma_i\}$. We denote by $\Omega = \Omega_1 \cap \ldots \cap \Omega_t$ and $\bar{\Omega} = \mathbb{Z}^*_n \setminus \Omega$. Let A be a PPT algorithm that returns 1 on input (x,n) if $x \in \Omega$. We define the advantage*

$$ADV_A^{\textit{SP-PS}}(\lambda) = \left| Pr[A(x,n) = 1 | x \xleftarrow{\$} \Omega] - Pr[A(x,n) = 1 | x \xleftarrow{\$} \bar{\Omega}] \right|.$$

The Strong Prime p-Subgroups assumption states that for any PPT algorithm A, the advantage $ADV_A^{PS}(\lambda)$ is negligible.

7 Implementation and Performance Analysis

7.1 Parameter Selection

The fastest currently known algorithm for factoring composite numbers is the Number Field Sieve (NFS) [14]. The expected running time of the NFS depends on the size of the modulus n and not on the size of its factors. More precisely, the expected running time is approximately

$$L[n] = e^{1.923 (\log n)^{1/3} (\log \log n)^{2/3}}.$$

In [13,14], the authors extrapolate the running time needed to factor a modulus of size λ_n from the computational effort required to factor a 512-bit modulus. Hence, a λ_n-bit modulus offers a security equivalent to a block cipher of d-bit security if

$$L[2^{\lambda_n}] \simeq 50 \cdot 2^{d-56} \cdot L[2^{512}]. \quad (1)$$

Since we start from a secure Okamoto-Uchiyama PKE and we want to optimize decryption by decreasing the size of some of the factors of the modulus, while keeping the size of the modulus constant, the NFS cannot be expected to factor n. Unfortunately, this strategy can make the resulting PKEs vulnerable to the Elliptic Curve Method (ECM) [11], if we lower the size of the factors below a certain threshold. Compared to the NFS, the ECM has the running time determined by the size of the smallest factor. Thus, if p is the smallest factor, then the running time of the ECM is

$$E[n,p] = (\log_2 n)^2 e^{\sqrt{2 \log p \log \log p}}.$$

Similarly to the NFS, Lenstra [12] extrapolates the equivalent security provided by a module of size λ_n with the smallest prime of size λ_p to be

$$E[2^{\lambda_n}, 2^{\lambda_p}] \geq 80 \cdot 2^{d-56} \cdot E[2^{768}, 2^{167}]. \quad (2)$$

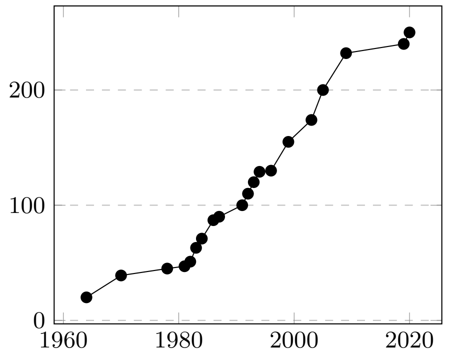

Fig. 1: Size of general number factored versus year

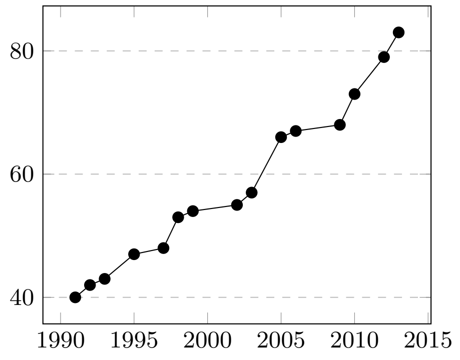

Fig. 2: Size of general number factored using ECM versus year

From Equations (1) and (2) we can deduce the following equivalency

$$E[2^{\lambda_n}, 2^{\lambda_p}] \geq 80 \cdot 2^{\log_2(L[2^{\lambda_n}]/(50 \cdot L[2^{512}]))} \cdot E[2^{768}, 2^{167}]. \tag{3}$$

A different model for predicting the security against the NFS and the ECM is provided in [7]. Compared to Lenstra's model, Brent uses known historical factoring records to predict the year a modulus of a given size will be factored. Using the least-squares fit, Brent obtains the following equation for the NFS

$$D_n^{1/3} = \frac{Y - 1928.6}{13.24} \text{ or equivalently } Y = 13.24 \cdot D_n^{1/3} + 1928.6 \tag{4}$$

and for the ECM

$$D_p^{1/2} = \frac{Y - 1932.3}{9.3} \text{ or equivalently } Y = 9.3 \cdot D_p^{1/2} + 1932.3, \tag{5}$$

where D_n is the number of digits of the factored modulus and D_p is the number of digits of the largest prime factor found using the ECM.

Using regression analysis we update Brent's equations using data points from [2, 18, 25] for the NFS and from [6, 27] for the ECM. These data points are presented in Figures 1 and 2.

Therefore, the updated equation for the NFS is

$$D_n^{1/3} = \frac{Y - 1926}{13.97} \text{ or equivalently } Y = 13.97 \cdot D_n^{1/3} + 1926 \tag{6}$$

and for the ECM

$$D_p^{1/2} = \frac{Y - 1939}{8.207} \text{ or equivalently } Y = 8.207 \cdot D_p^{1/2} + 1939. \tag{7}$$

Equations (4) to (7) are presented in Figures 3 and 4. Note that the black dots represent the acquired data points. We can see that in the case of the

Modulus key size	3072	7680	15360
Lenstra model	800(3)	1617(4)	2761(5)
Regression model	749(4)	1457(5)	2385(6)

Table 2: Equivalent key sizes

NFS the estimates are close, while in the case of the ECM the new estimate is more pessimistic from a security point of view. Using the updated estimates (Equations (6) and (7)) we obtain the following equivalency

$$D_p^{1/2} = \frac{13.97 \cdot D_n^{1/3} - 13}{8.207}. \tag{8}$$

According to NIST [5], the recommended key sizes for composite modules are $\lambda_n = 3072/7680/15360$. We preferred to use NIST recommendations instead of the ones from [13, 14] since these key sizes are the ones used by the industry and the key sizes from [13, 14] are criticized as being to conservative [25]. Therefore, using Equations (3) and (8) we obtain the equivalent size of the smallest prime. The results are presented in Table 2. Note that in parenthesis we provide the maximum number of prime factors that n can have. Based on these equivalences, we obtain the parameters for the Okamoto-Uchiyama schemes that offer protection against the NFS and the ECM (see Table 3). We can see that the only key sizes that support a multiprime version are 7680 in the regression model and 15360 in both models.

7.2 Complexity

Using the complexities provided in Table 1, we computed the asymptotic run times of the decryption algorithm for each Okamoto-Uchiyama variant. We also

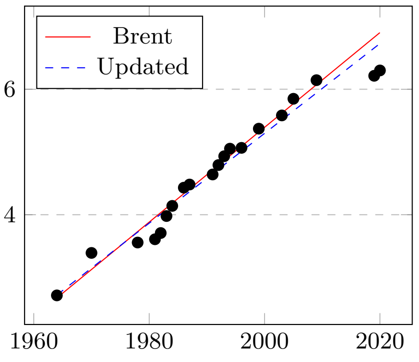

Fig. 3: $D^{1/3}$ versus year Y

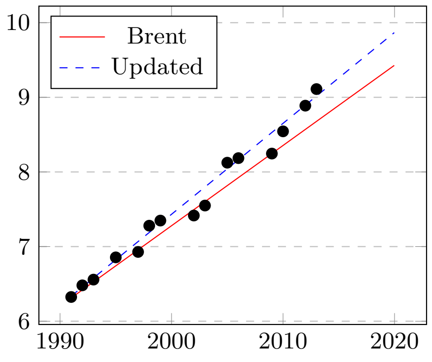

Fig. 4: $D^{1/2}$ versus year Y for ECM

$\|n\|$	t	$\|p\|$	$\|q\|$	Model	Type
3072	1	1024	1024	-	Balanced
	1	800	1472	Lenstra	Unbalanced
	1	749	1574	Regression	Unbalanced
7680	1	2560	2560	-	Balanced
	1	1617	4446	Lenstra	Unbalanced
	1	1457	4766	Regression	Unbalanced
	2	1457	1852	Regression	Multiprime
15360	1	5120	5120	-	Balanced
	1	2761	9838	Lenstra	Unbalanced
	1	2385	10590	Regression	Unbalanced
	2	2761	4316	Lenstra	Multiprime
	2	2385	5820	Regression	Multiprime

Table 3: Okamoto-Uchiyama parameters' size

determined the size of the message space for each variant. The results are provided in Table 4. Note that by parallel multiprime we mean the multiprime version in which we use two separate threads to compute m_1 and m_2.

Scheme	Decryption Complexity	$\|m\|$
Balanced	$\mathcal{O}(2\lambda M(2\lambda) + \lambda M(\lambda))$	$\lambda - 1$
Unbalanced	$\mathcal{O}(2\lambda_p M(2\lambda_p) + \lambda_p M(\lambda_p))$	$\lambda_p - 1$
Multiprime	$\mathcal{O}(2t\lambda_p M(2\lambda_p) + t\lambda_p M(\lambda_p) + \log t M(t\lambda_p))$	$\lambda_p t - 1$
Parallel Multiprime	$\mathcal{O}(2\lambda_p M(2\lambda_p) + \lambda_p M(\lambda_p) + \log t M(t\lambda_p))$	$\lambda_p t - 1$

Table 4: Performance analysis

The comparison of the computational complexity of the four variants is presented in Figures 5 and 6. Note that we only provide the comparison for $\lambda_n = 7680/15360$ since these are the only module sizes that support all variants. In the case of $\lambda_n = 3072$, we can easily deduce that the unbalanced version (UnB) will run faster than the balanced one (Bal). Note that in Figure 6 the two dots for each multiprime version (Mp) correspond to the two equivalence models: Lenstra - upper dot and Regression - lower dot.

From the two plots we can see that the parallel multiprime version (PMp) has a running time comparable to the unbalance version, and since the multiprime version has a larger message space, it should be preferred. Nevertheless, if only one thread is available and messages are below a certain threshold (denoted by dotted lines), then the unbalanced version is preferred, otherwise the multiprime version should be used.

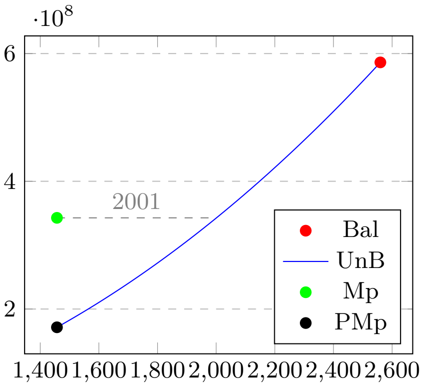

Fig. 5: Decryption complexity versus λ_p for $\lambda_n = 7680$

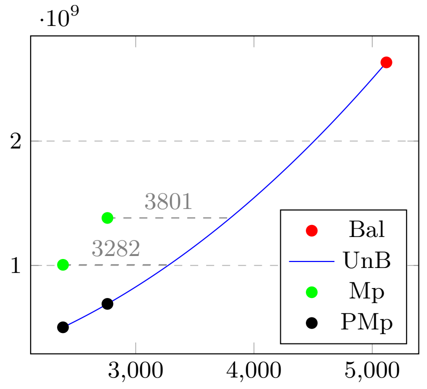

Fig. 6: Decryption complexity versus λ_p for $\lambda_n = 15360$

7.3 Implementation Details

We further provide the reader with benchmarks for the four Okamoto-Uchiyama PKE schemes. Note that besides the parameters from Table 3, we also used the parameters from Table 5 (see Figures 5 and 6).

$\|n\|$	t	$\|p\|$	$\|q\|$	Model	Type
7680	1	2001	3678	Regression	Unbalanced
15360	1	3801	7758	Lenstra	Unbalanced
	1	3282	8796	Regression	Unbalanced

Table 5: Additional Okamoto-Uchiyama parameters' size

We ran each of the three sub-algorithms on a CPU Intel i7-4790 4.00 GHz and used GCC to compile it (with the O3 flag activated for optimization). Note that for all computations we used the GMP library [3]. To calculate the running times we used the *omp_get_wtime()* function [1]. For the parallel multiprime variant we used the OMP library [1] to parallelize decryption. To obtain the average running time in seconds we chose to encrypt 100 128/192/256-bit messages. Therefore, we wanted to simulate a key distribution scenario.

The results are provided in Table 6. Note that with blue we marked the additional parameters and the decryption times for the parallel multiprime version are given in the second row of $t = 2$. Also, the optimized version of the decryption algorithm is denoted by *Decrypt* (opt).

In Table 6 we omitted encryption times, since they are similar to each other for a given λ_n. More precisely, for 3072 we obtain 0.009679, for 7680 we have 0.096855 and for 15360 we have 0.550525. We can see from Table 6 that the conclusions from Section 7.2 hold. We can also see that the multiprime version has the shortest time to generate the parameters, while the unbalanced version the longest time. Nevertheless, generating parameters is a one-time operation.

$\lvert n\rvert$	t	$\lvert p\rvert$	$\lvert m\rvert$	*Setup*	*Decrypt*	*Decrypt* (opt)
3072	1	1024	1023	0.058187	0.003073	0.001537
	1	800	799	0.120665	0.001535	0.000772
	1	749	748	0.121719	0.001231	0.000614
7680	1	2560	2559	1.696050	0.034409	0.017205
	1	1617	1616	5.828290	0.010711	0.005353
	1	1457	1456	6.644670	0.007872	0.003932
	1	2001	2000	3.158020	0.019550	0.009751
	2	1457	2913	0.476329	0.015782	0.007897
					0.008538	0.004511
15360	1	5120	5119	17.04490	0.203512	0.101740
	1	2761	2760	70.11920	0.040647	0.020362
	1	3801	3800	37.98000	0.089592	0.044745
	1	2385	2384	103.1440	0.031670	0.015889
	1	3282	3281	44.25260	0.063980	0.032011
	2	2761	5521	7.584640	0.081344	0.040684
					0.041635	0.021137
	2	2385	4769	15.86700	0.063377	0.031778
					0.032505	0.016557

Table 6: Message size and running times

8 Conclusions

In this work we introduced two novel versions of the Okamoto-Uchiyama cryptosystem. The first one, called the unbalanced Okamoto-Uchiyama PKE, lowers the size of p in order to decrease decryption time at the cost of shrinking the message space. The second one, called the multiprime Okamoto-Uchiyama PKE, increases the number of factors and achieves a decryption time comparable with the unbalanced version if multiple threads are available. An advantage of the multiprime variant is that it has a larger message space than the unbalanced version and, sometimes, even larger than the original PKE. Therefore, if parallel threads are available we recommend the multiprime version due to its larger message space, otherwise, if short messages are sent we recommend the unbalanced variant.

We also argue why we did not compare our variants to the Coron-Naccache-Paillier PKE. Since equivalence to factoring is not proven in the original paper and there are doubts about why simply modifying the proof of the Okamoto-Uchiyama PKE does not work, we chose to discuss the Coron-Naccache-Paillier PKE separately from the Okamoto-Uchiyama variants. For completeness, in Appendix A we provide an unbalanced and a multiprime variant of the Coron-Naccache-Paillier PKE, study their security and provide performance benchmarks. As in the case of the Okamoto-Uchiyama PKE, we recommend the unbalance version for short messages when decryption parallelization is not possible, and the multiprime one otherwise.

Future work. We leave for future work the adaptation to unbalanced and multiprime mode, as well as the performance comparison with other factoring related cryptosystems such as Paillier [20] or RSA [4, 22, 24].

References

1. OpenMP. https://www.openmp.org/
2. RSA Factoring Challenge. https://en.wikipedia.org/wiki/RSA_Factoring_Challenge
3. The GNU Multiple Precision Arithmetic Library. https://gmplib.org/
4. Cryptography Using Compaq Multiprime Technology in a Parallel Processing Environment. https://www.compaq.com (2000)
5. Barker, E.: NIST SP800-57 Recommendation for Key Management, Part 1: General. Tech. rep., NIST (2020)
6. Brent, R.P.: Large Factors Found By ECM. https://maths-people.anu.edu.au/~brent/ftp/champs.txt
7. Brent, R.P.: Some Parallel Algorithms for Integer Factorisation. In: Euro-Par 1999. Lecture Notes in Computer Science, vol. 1685, pp. 1–22. Springer (1999)
8. Choi, D.H., Choi, S., Won, D.: Improvement of Probabilistic Public Key Cryptosystems Using Discrete Logarithm. In: ICISC 2001. Lecture Notes in Computer Science, vol. 2288, pp. 72–80. Springer (2001)
9. Coron, J., Naccache, D., Paillier, P.: Accelerating Okamoto-Uchiyama Public-Key Cryptosystem. Electronics Letters **35**(4), 291–292 (1999)
10. Crandall, R., Pomerance, C.: Prime Numbers: A Computational Perspective. Number Theory and Discrete Mathematics, Springer (2005)
11. Jr., H.W.L.: Factoring Integers with Elliptic Curves. Annals of Mathematics pp. 649–673 (1987)
12. Lenstra, A.K.: Unbelievable Security. Matching AES Security Using Public Key Systems. In: ASIACRYPT 2001. Lecture Notes in Computer Science, vol. 2248, pp. 67–86. Springer (2001)
13. Lenstra, A.K., Verheul, E.R.: Selecting Cryptographic Key Sizes. In: PKC 2000. Lecture Notes in Computer Science, vol. 1751, pp. 446–465. Springer (2000)
14. Lenstra, A.K., Verheul, E.R.: Selecting Cryptographic Key Sizes. Journal of Cryptology **14**(4), 255–293 (2001)
15. Lim, S., Kim, S., Yie, I., Lee, H.: A Generalized Takagi-Cryptosystem with a modulus of the form $p^r q^s$. In: INDOCRYPT 2000. Lecture Notes in Computer Science, vol. 1977, pp. 283–294. Springer (2000)
16. Maimuţ, D., Teşeleanu, G.: A New Generalisation of the Goldwasser-Micali Cryptosystem Based on the Gap 2^{k}-Residuosity Assumption. In: SecITC 2020. Lecture Notes in Computer Science, vol. 12596, pp. 24–40. Springer (2020)
17. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC press (1996)
18. Odlyzko, A.M.: The Future of Integer Factorization. RSA Laboratories Cryptobytes **1**(2), 5–12 (1995)

19. Okamoto, T., Uchiyama, S.: A New Public-Key Cryptosystem as Secure as Factoring. In: EUROCRYPT 1998. Lecture Notes in Computer Science, vol. 1403, pp. 308–318. Springer (1998)
20. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: EUROCRYPT 1999. Lecture Notes in Computer Science, vol. 1592, pp. 223–238. Springer (1999)
21. Pei, D., Salomaa, A., Ding, C.: Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography. World Scientific Publishing (1996)
22. Rivest, R.L., Shamir, A., Adleman, L.M.: Cryptographic Communications System and Method (1983), US Patent 4,405,829
23. Sakurai, K., Takagi, T.: On the Security of a Modified Paillier Public-Key Primitive. In: ACISP 2002. Lecture Notes in Computer Science, vol. 2384, pp. 436–448. Springer (2002)
24. Shamir, A.: RSA for Paranoids. RSA Laboratories Cryptobytes **1**(3), 1–4 (1995)
25. Silverman, R.D.: A Cost-Based Security Analysis of Symmetric and Asymmetric Key Lengths. RSA Laboratories' Bulletin 13 (2000)
26. Teşeleanu, G.: The Case of Small Prime Numbers Versus the Joye–Libert Cryptosystem. Mathematics **10**(9), 1577 (2022)
27. Zimmermann, P.: 50 Largest Factors Found by ECM. https://members.loria.fr/PZimmermann/records/top50.html

A The Multipleprime Coron-Naccache-Paillier PKE scheme

A.1 Preliminaries

Before presenting the multiprime generalization, we first introduce the generalized CRT as stated in [21].

Theorem 3 (Generalized Chinese Remainder Theorem). *Let $m_1, m_2, \ldots, m_t$ be positive integers. For a set of integers $a_1, a_2, \ldots, a_t$ the system of congruences*

$$x \equiv a_i \pmod{m_i},\ \textit{for } i \in [1, t]$$

has solutions if and only if

$$a_i \equiv a_j \pmod{\gcd(m_i, m_j)},\ \textit{for } i \neq j,\ i, j \in [1, t]. \tag{9}$$

If Equation (9) holds, then the solution will be unique modulo $\operatorname{lcm}(m_1, m_2, \ldots, m_t)$.

A.2 Description

For completeness, we further describe the multiple prime generalisation of the Coron-Naccache-Paillier optimisation. Bear in mind that when $t = 1$ and $\lambda_p = \lambda_q$ we obtain the optimisation presented in [9] and when $t = 1$ we obtain the unbalanced version of the Coron-Naccache-Paillier scheme. The reader can easily see that we can also use the optimisation techniques presented in Section 5 to speed-up this proposal.

Setup$(\lambda_p, \lambda_q, \lambda_u)$: Generate t distinct large prime numbers $u_1, \ldots, u_t$ such that $|u_1| = \ldots = |u_t| = \lambda_u$. Also, generate $t+1$ distinct large prime numbers $p_1, \ldots, p_t, q$ such that $|p_1| = \ldots = |p_t| = \lambda_p$, $|q| = \lambda_q$ and $p_i - 1 = u_i v_i$, for $i \in [1, t]$. Let $n = p_1^2 \cdot \ldots \cdot p_t^2 q$. Randomly select $g \in \mathbb{Z}_n$ such that for any $i \in [1, t]$ the element $g_{p_i} \equiv g^{p_i - 1} \bmod p_i^2$ has order p_i in $\mathbb{Z}^*_{p_i^2}$. Let $\ell = \text{lcm}(p_1 u_1 v_1, \ldots, p_t u_t v_t)$. Using the generalized CRT compute the unique $v \in \mathbb{Z}_\ell$ such that $v \equiv v_i \bmod p_i u_i v_i$. Compute $G \equiv g^v \in \mathbb{Z}_n^*$ and $H \equiv G^n \bmod n$. Output the public key $pk = (n, G, H, \lambda, t)$ and the corresponding secret key $sk = (P, q)$, where $P = \{p_i\}_{i \in [1,t]}$.

Encrypt(pk, m): To encrypt a message $m \in [0, 2^{\lambda_u t - 1})$, we choose $r \xleftarrow{\$} \mathbb{Z}_n$ and compute $c \equiv G^m H^r \bmod n$. Output the ciphertext c.

Decrypt(sk, c): For each $i \in [1, t]$ compute $c_{p_i} \equiv c^{u_i} \bmod p_i^2$, $g_{p_i} \equiv G^{u_i} \bmod p_i^2$ and recover m_i from the relation

$$m_i \equiv \frac{L(c_{p_i})}{L(g_{p_i})} \bmod p_i.$$

Let $n' = p_1 \cdot \ldots \cdot p_t$. Using the CRT compute the unique $m \in \mathbb{Z}_{n'}$ such that $m \equiv m_i \bmod p_i$.

Correctness. The first thing we need to show is that in the *Setup* phase the conditions of Theorem 3 are satisfied. We note that

$$\gcd(p_i u_i v_i, p_j u_j v_j) = \gcd(v_i, v_j) \mid v_i - v_j,$$

and therefore v exists.

To recover m we only need to prove that we end up with the same relations as in the case of the Okamoto-Uchiyama decryption algorithm. Therefore, we have

$$c_{p_i} \equiv c^{u_i} \equiv (G^m H^r)^{u_i} \equiv (g^m h^r)^{v u_i} \equiv (g^m h^r)^{p_i - 1} \bmod p_i^2,$$

since $v \equiv v_i \bmod p_i(p_i - 1)$.

A.3 Security Analysis

Theorem 4. *The multiple prime Coron-Naccache-Paillier PKE is* IND-CPA *secure if and only if the* SP-PS *assumption is intractable.*

Proof (sketch). We further denote by $E(m)$ the encryption of a message m. Note that breaking the SP-PS assumption is equivalent with distinguishing between $E(0)$ and $E(1)$. To see that we first compute

$$E(0)^{u_i} \equiv (G^0 H^{r_0})^{u_i} \equiv G^{r_0 n u_i} \equiv g^{r_0 n (p_i - 1)} \equiv 1 \bmod p_i^2,$$

and

$$E(1)^{p_i - 1} \equiv (G^1 H^{r_1})^{u_i} \equiv G^{u_i} G^{r_1 n u_i} \equiv g^{p_i - 1} g^{r_1 n (p_i - 1)} \equiv g^{p_i - 1} \bmod p_i^2.$$

Scheme	Decryption Complexity
Balanced	$\mathcal{O}(2\lambda_u M(2\lambda) + \lambda M(\lambda))$
Unbalanced	$\mathcal{O}(2\lambda_u M(2\lambda_p) + \lambda_p M(\lambda_p))$
Multiprime	$\mathcal{O}(2t\lambda_u M(2\lambda_p) + t\lambda_u M(\lambda_p) + \log tM(t\lambda_p))$
Parallel Multiprime	$\mathcal{O}(2\lambda_u M(2\lambda_p) + \lambda_u M(\lambda_p) + \log tM(t\lambda_p))$

Table 7: Performance analysis

This implies that the order of $E(0)^{u_i} \bmod p_i^2$ divides u_i and the order of $E(1)^{u_i} \bmod p_i^2$ is p_i since $g^{p_i-1} \bmod p_i^2$ has order p_i. Therefore, $E(0)^{u_i} \in \bar{\Omega}$ and $E(1)^{u_i} \in \Omega$.

Now, let's assume that there exists a PPT adversary A that can distinguish between $E(0)$ and $E(1)$. We further construct another PPT algorithm B that can break the IND-CPA security of our proposal.

Once B receives a ciphertext C, he computes the value $X \equiv CG^{-m_0} \bmod n$ and chooses $r \xleftarrow{\$} \mathbb{Z}_n$. Let $\bar{G} \equiv G^{(m_1-m_0)+rn} \bmod n$, $\bar{H} \equiv \bar{G}^n \bmod n$ and $\ell = \mathrm{lcm}(p_1 - 1, \ldots, p_t - 1, q - 1)$. We denote by $\bar{E}(m)$ the encryption of m using $\bar{G}$ and $\bar{H}$. If B receives the encryption of m_0, we have $X = \bar{E}(0)$ and $X = \bar{E}(1)$, otherwise.

To prove that $\bar{G}^{u_i} \bmod p_i^2$ has order p_i for all $i \in [1, t]$, we first observe that $\bar{G}^{u_i} \equiv \bar{g}^{p_i-1}$, where $\bar{g} \equiv g^{(m_1-m_0)+rn} \bmod n$. Using the same arguments as in the proof of Theorem 2 we obtain that $\bar{G}^{u_i} \in \Gamma_i$, and thus $\bar{G}^{u_i} \in \Omega_i$.

On input $(\bar{G}, \bar{H}, X)$, adversary A outputs the correct bit b with non-negligible probability. Algorithm B simply relays b and according to the arguments presented above, it guesses correctly whether c is the encryption of m_0 or m_1.

We further prove the converse statement. Thus, let's assume that there exists a PPT adversary A that guesses correctly if a ciphertext encrypts either m_0 or m_1. We assume without loss of generality that $m_0 < m_1$. We will construct a PPT machine B which can distinguish between $E(0)$ and $E(1)$.

Given a ciphertext c, algorithm B constructs an element $X \equiv C^{m_1-m_0}G^{m_0+nr}$, where $r \xleftarrow{\$} \mathbb{Z}_n$. If B receives an encryption of 0 we have $X = E(m_0)$ and $X = E(m_1)$, otherwise.

On input (G, H, X), adversary A outputs a bit b. Therefore, if algorithm B outputs b, then with non-negligible probability it guesses correctly whether C is the encryption of 0 or 1. $\square$

A.4 Complexity and Implementation Details

Similarly to Table 4, we computed the complexity of the decryption algorithm for the Coron-Naccache-Paillier variants. The results can be seen in Table 7. Note that the message size for the balanced and unbalanced variants is $\lambda_u - 1$ and for the multiprime versions is $\lambda_u t - 1$. Compared to the Okamoto-Uchiyama, the Coron-Naccache-Paillier variants have faster decryption times, but at the cost of a smaller message space and the loss of equivalence with factoring.

According to [9, 14], λ_u should be chosen such that the subgroup generated by G is large enough that it protects the PKE against the baby-step-giant-step method. According to [5], we should choose $\lambda_u = 128/192/256$. However, we want to simulate a key distribution scenario, and thus we have to choose $\lambda_u = 129/193/257$. This implies that the message space is 128/192/256 for the (un)balanced version and 257/385/513 for the multiprime ones. We further provide the benchmarks for the Coron-Naccache-Paillier versions in Table 8.

$\|n\|$	t	$\|p\|$	*Setup*	*Decrypt*	*Decrypt* (opt)
3072	1	1024	0.063806	0.000436	0.000215
	1	800	0.121260	0.000278	0.000139
	1	749	0.147123	0.000240	0.000119
7680	1	2560	1.481400	0.002870	0.001431
	1	1617	4.954550	0.001399	0.000697
	1	1457	8.247510	0.001149	0.000573
	2	1457	0.570016	0.002308	0.001162
				0.001692	0.000916
15360	1	5120	16.67890	0.011248	0.005606
	1	2761	76.39660	0.004128	0.002063
	1	2385	112.3960	0.003715	0.001856
	2	2761	7.892090	0.008284	0.004155
				0.004840	0.002749
	2	2385	11.83500	0.007450	0.003733
				0.004308	0.002429

Table 8: Running times

Based on Tables 7 and 8, we can see that the parallel multiprime version has a running time comparable to the unbalance version, and since the multiprime version has double the message space, it should be preferred. Nevertheless, if only one thread is available and we only want to distribute symmetric keys, then the unbalanced version is preferable, otherwise the multiprime version should be used.