Homomorphic public-key systems based on subgroup membership problems

Kristian Gjøsteen

September 15, 2003

Abstract

We describe the group structure underlying several popular homomorphic public-key systems and the problems they are based on. We prove several well-known security results using only the group structure and assumptions about the related problems.

Then we provide examples of two new instances of this group structure and analyse their security.

1 Introduction

A cryptosystem is homomorphic with respect to some operation $*$ on the message space if there is a corresponding operation $*'$ on the ciphertext space such that for encryptions c, c' of messages m, m', $c *' c'$ is an encryption of $m * m'$.

Goldwasser and Micali [7] introduced the concept of semantic security and described a semantically secure cryptosystem based on the quadratic residue assumption. It was later remarked that this cryptosystem really was homomorphic with respect to addition in $\mathbb{Z}_2$. Naccache and Stern [8] introduced an homomorphic cryptosystem based on "higher residues" in $\mathbb{Z}_n^*$, where n was a product of two primes of a special form. Okamoto and Uchiyama [9] described an homomorphic cryptosystem over $\mathbb{Z}_n^*$ using a modulus of the form $n = p^2 q$.

Paillier [10] introduced an homomorphic cryptosystem based on the ring $\mathbb{Z}_{n^2}$, where n was simply an RSA modulus. This was generalised by Damgård and Jurik [3], who also adapted Shoup's RSA threshold trick and applied it to electronic voting.

From a group perspective, all of these systems are essentially the same. In Section 2, we shall give a generic description and make precise exactly what the conditions on the group are. In Section 3 we show what extra conditions are required for threshold decryption as in [3]. In Section 4, we describe several cryptosystems of this form.

We discuss security in Section 5. The main result is that the system is semantically secure against chosen plaintext attacks under a certain assumption on the group. We discuss the digit security of the system and comment briefly on security against chosen ciphertext attack models.

In Section 6 we present a modification of the Naccache-Stern system to satisfy the requirement of threshold decryption. In Section 7 we describe how some of the variants can be combined to gain a small increase in bandwidth.

Sections 2–5 is mostly well-known, but adapted to the more general group-theoretic setting. We believe the material in Section 6 and 7 is new.

2 The group structure

The group G should be abelian and have two subgroups H and K such that $H \cap K = \{1\}$ and $HK = G$. We require H to be cyclic and $\gcd(|H|, |K|) = 1$.

The basic idea is that H should hold the message, and K should act as a cloak, hiding the message. We embed a message $m \in \{0, \dots, |H| - 1\}$ simply by raising to the power m a public generator of H. To randomise, we choose a random element of K and multiply the two elements.

So it must be easy to compute discrete logarithms in the subgroup H, that is, we need an easily computable isomorphism $\log_H : H \to \mathbb{Z}_{|H|}$. We will sometimes consider $\log_H$ as just a map from H into the set $\{0, \dots, |H| - 1\}$.

We also need a probabilistic polynomial-time algorithm σ that takes as input a bit string of length k and outputs an element of K with negligible probability of failure. We require that the distribution of the output should be computationally indistinguishable from the uniform distribution, so that we can choose group elements almost uniformly at random by choosing bit strings uniformly at random.

Remark. A general solution is to make public an element $g_K \in K$, and let $\sigma(r) = g_K^r$. In this case, we may as well assume that g_K generates K.

If there is an algorithm σ', taking bit strings of length k as input and outputting elements of G almost uniformly distributed, one can let $\sigma(r) = (\sigma'(r))^{|H|}$, which will be a random element of K. Here, we allow σ' to output something which is not in G with negligible probability.

Key generation The input is a security parameter τ. A group G is chosen satisfying the above requirements and matching τ, along with an element $g \in G$ such that $H \subseteq \langle g \rangle$.

Second, using the Chinese Remainder Theorem, a number d is computed such that

$$d \equiv \begin{cases} 0 \pmod{|K|} \\ 1 \pmod{|H|} \end{cases}$$

The public information is (G, g, σ). As discussed above, the description of σ can consist for example of a group element g_K, or of the group order $|H|$

and a description of σ'. The private information is simply d. The number $|K|$ should be kept secret (it can be discarded since it is no longer required).

Remark. Note that $H \times K$ is isomorphic to G and the isomorphism is $(h, k) \mapsto hk$. For any element $x \in G$, there exists $h \in H$, $k \in K$ such that $x = hk$ and $x^d = h^d k^d = h$. So the map $x \mapsto x^d$ is simply the projection on H.

Encryption The input is $m \in \{0, \ldots, |H| - 1\}$. Choose a random k-bit string r. The encryption function $E : \{0, \ldots, |H| - 1\} \times \{0, 1\}^k \to G$ is $E(m, r) := g^m \sigma(r)$.

Decryption The input is a cipher text $c \in G$. The message recovery function $D : G \to \{0, \ldots, |H| - 1\}$ is $D(c) := \log_H(c^d)$.

We now see that $\log_H(g^d)$ must be 1, which fixes the isomorphism. Correctness is then obvious when we note that for any $g \in G$,

$$\log_H(c^d) = \log_H(\sigma(r)^d (g^m)^d) = \log_H((g^d)^m) = m.$$

If we consider the set $\{0, \ldots, |H| - 1\}$ as representatives for the elements of $\mathbb{Z}_{|H|}$, the system is homomorphic with respect to addition in $\mathbb{Z}_{|H|}$.

The security of the cryptosystem based on this group structure depends on the following problems.

Definition. Let G be an abelian group with subgroups K, H such that $G = KH$ and $K \cap H = \{1\}$. Let $\pi : G \to H$ denote the projection onto H.

The *subgroup projection problem* (SPP) is the following: For an abelian group $G = KH$, $K \cap H = \{1\}$, find the projection of an element $g \in G$ onto H.

The *subgroup discrete logarithm problem* (SDLP) is the following: For two elements of G, g and c, find the discrete logarithm of $\pi(c)$ to the base $\pi(g)$.

The *decisional subgroup membership problem* (DSMP) is the following: For an abelian group G with a subgroup K, the problem is to decide whether a given $g \in G$ is in K or not. We denote this DSMP as (G, K).

Remark. Yamamura and Saito [13] defines the DSMP problem and notes that the semantic security of the cryptosystems described in Section 4 is based on the DSMP.

Remark. The Decision Diffie-Hellman problem (DDHP) is often characterised as a decision subgroup membership problem. If g is a generator of G, then given g^x, g^y and g^z the DDHP is to say if $g^z = g^{xy}$. This is equivalent to asking if (g^x, g^z) is in the subgroup of $G \times G$ generated by (g, g^y).

Note that this is a very different DSMP from our DSMP. We ask a question about a subgroup of a cyclic group, while the DDHP asks a question about a subgroup of a non-cyclic group.

Remark. These problems can also be phrased in terms of residue classes: The SPP is to find a "canonical" representative for the residue class of c in G/K. The SDLP is to find the discrete logarithm of cK to the base gK. The DSMP is to decide if the residue class cK is the same as the residue class $1K$, that is, if c is an $|H|$th residue.

For the group G, it is clear that DSMP is easier than the SDLP, which in turn is easier than the SPP.

If $|K|$ and $|H|$ are known, the SPP is trivial, so obviously $|K|$ must be kept secret. Strictly speaking, $|H|$ need not be made public. Only an upper bound on the length of messages is needed to encrypt messages correctly. However, care must be taken to prevent the group order leaking. If $|H|$ is made public, it must still be difficult to compute $|K|$.

For some of the concrete instances describe in Section 4, the SPP and the SDLP are equivalent. This is true when a generator of H can be made public.

It is clear that the scheme is one-way if and only if the SDLP is hard. In Section 5 we shall show that the scheme is semantically secure if and only if the DSMP is hard.

Remark. We have for some d' that $d \equiv d'|K| \pmod{|K||H|}$. Notice that

$$\log_H(c^d) = \log_H((c^{|K|})^{d'}) = d'\log_H(c^{|K|});$$

This will certainly be more efficient for decryption (since $|K|$ is typically smaller than d), but it is inappropriate for some applications.

Remark. ElGamal is homomorphic with respect to multiplication, that is, the product of two ciphertexts contains the product of the messages. In general, one wants the system to be homomorphic with respect to addition.

Damgård, Groth and Salomonsen [4] note that if messages are restricted to some subgroup where discrete logarithms are easy to compute, but the corresponding subgroup discrete logarithm problem is difficult, the system becomes homomorphic with respect to addition. And the group G with messages in H is an obvious candidate.

3 A threshold variant

We briefly outline how the decryption work can be shared. Recall that the main part of the decryption procedure is computing a dth power.

Shamir's secret sharing scheme [11] works roughly as follows: We construct a polynomial $f(x) = d + \sum_{i=1}^{t-1} f_i x_i$, where d is the exponent we want to compute, and $f_1, \ldots, f_{t-1}$ are random integers from the set $\{0, \ldots, |G|-1\}$. We construct l shares $x_1, \ldots, x_l$ as $x_i := f(i)$.

To compute the dth power of the cipher text c, each share owner computes $c_i = c^{x_i}$. Shares are combined using Lagrange interpolation to recover the value $f(0) = d$ in the exponent. If S is a subset of $\{1, \ldots, l\}$, $|S| = t$, then with

$$\lambda_i = \prod_{\substack{i' \in S \\ i' \neq i}} \frac{-i}{i - i'}$$

we get

$$\prod_{i \in S} c_i^{\lambda_i} = c^{\sum_{i \in S} f(i) \prod_{\substack{i' \in S \\ i' \neq i}} \frac{-i}{i-i'}} = c^d.$$

The obvious problem in our case is that we cannot compute $(i - i')$th roots in the group G.

Shoup [12] gave a small trick that we can use to avoid this problem. With $\Delta = l!$, we note that $\Delta\lambda_i \in \mathbb{Z}$ and that

$$\prod_{i \in S} c_i^{\Delta\lambda_i} = c^{\Delta d}.$$

To simplify the security proof, each share is computed as $c^{\Delta x_i}$, so the actual computational result is $c^{\Delta^2 d}$. If $\gcd(\Delta, |H|) = 1$, we can of course still recover the message simply by multiplying with Δ^{-2} modulo $|H|$.

The extra requirements on G are then that $\gcd(|H|, \Delta) = 1$, and that public knowledge of elements of H does not compromise the security.

One application for such threshold systems is for electronic voting systems, as described in [3].

4 Known instances

The systems described in this section are essentially the same as the original systems, but some small changes may have been introduced to simplify the exposition. We also note that this list is not intended to be complete, but to describe the basic approaches.

Goldwasser-Micali [7] Let v_1, v_2 be odd numbers each containing a large prime, and such that $p_1 = 2v_1 + 1$, $p_2 = 2v_2 + 1$ both are prime. Let $n = p_1 p_2$, K be the set of quadratic residues in $\mathbb{Z}_n^*$, and let H be the subgroup $\{1, -1\}$. The group KH is then simply the subgroup of elements in $\mathbb{Z}_n^*$ having Jacobi symbol 1.

We have $|H| = 2$, $|K| = v_1 v_2$ and $\sigma'(r) = r^2$. The public key is simply n. Recovering the message can be done by reducing modulo one of the primes and computing the Jacobi symbol, but of course, computing the v_1v_2th power also works.

In this case, the DSMP is the Quadratic Residue problem, determine if a number is a quadratic residue modulo composite numbers.

Naccache-Stern [8] Let u_1, u_2 be relatively prime B-smooth integers (with B small). Choose distinct primes v_1, v_2 such that $p_1 = 2v_1u_1 + 1, p_2 = 2v_2u_2 + 1$ are primes and $\gcd(v_1v_2, u_1u_2) = 1$, and let $n = p_1p_2$. (Taking v_1, v_2 to be primes simplifies the exposition. We could weaken this to say that each of them must contain a big prime factor.)

The group G is then the set of quadratic residues in $\mathbb{Z}_n^*$, K is the cyclic subgroup of order v_1v_2 and H is the cyclic subgroup of order u_1u_2. Now $g \in G$ must be a generator (if it does not generate K, it reveals a factor of n).

If one wants to keep $|H|$ secret, then g_K should be an element of order v_1v_2. Otherwise, with $k = \lceil \log_2 n \rceil$, $\sigma' : \{0,1\}^k \to G$ should be the function $\sigma'(r) = r^2$, where the bit string r as usual is interpreted as an integer.

Finding discrete logarithms in H to the base g^d is easy, since $|H|$ is a B-smooth number. B is then simply a bound on how much work one is willing to do to recover the message from the cipher.

In this setting, the DSMP is simply deciding if an element $c \in \mathbb{Z}_n^*$ is an $|H|$th power.

The knowledge of any element of H along with the order of H will allow anyone to factor the modulus. This means that the system is not usable as a threshold system. We present a small modification to improve on this in Section 6.

Okamoto-Uchiyama [9] Let p and q be primes such that $p \not| (q-1)$, and let $n = p^2q$. Then the order of $\mathbb{Z}_n^*$ is $p(p-1)(q-1)$ and $\mathbb{Z}_n^* \simeq \mathbb{Z}_{pq}^* \times \mathbb{Z}_p$. K is the subgroup of order $(p-1)(q-1)$, H is the subgroup of order p.

Obviously, one cannot publish $|H| = p$, so the system will be impossible to use as a threshold system. Any element g with order divisible by p can be used as generator (that is, $g^{(p-1)(q-1)} \neq 1$). We can use $\sigma'(r) = r^n \in K$, or publish such an element.

We briefly describe how to solve discrete logarithms in H: Considering the element $1 + p \in \mathbb{Z}_{p^2}$, we see that

$$(1+p)^m \equiv 1 + mp + \sum_{i=2}^{m} \binom{m}{i} p^i \equiv 1 + mp \pmod{p^2}.$$

This means that the discrete logarithm problem is easy in the subgroup of $\mathbb{Z}_{p^2}^*$ generated by $1 + p$.

The security of this cryptosystem is equivalent to factoring numbers on the form $n = p^2q$. We note that the public key is easy to simulate when you know that n has the required form, as almost all elements in $\mathbb{Z}_n$ are in $\mathbb{Z}_n^*$ and have order divisible by p. Given access to an oracle that can decrypt messages for the public key (n, g), we can compute the ciphertext g^z for some random number $z = O(n)$. The answer is a message m such that $z \equiv m \pmod{p}$. Most likely we will have $\gcd(z - m, n) = p$.

Paillier [10] Let $n = pq$ be an ordinary RSA modulus such that $\gcd(n, (p-1)(q-1)) = 1$. Then $\mathbb{Z}_{n^2}^*$ has order $(p-1)(q-1)n$ and $\mathbb{Z}_{n^2}^* \simeq \mathbb{Z}_n^* \times \mathbb{Z}_n$. K is then the part isomorphic to $\mathbb{Z}_n^*$ and H is the part isomorphic to $\mathbb{Z}_n$.

As above, we note that

$$(1+n)^m = 1 + mn + \sum_{i=2}^{m} \binom{m}{i} n^i \equiv 1 + mn \pmod{n^2},$$

so the discrete logarithm problem is easy in the subgroup generated by $1+n$.

The DSMP in this case is what Paillier terms the Decisional Composite Residue problem.

Damgård and Jurik [3] noticed that this system extends to $\mathbb{Z}_{n^{s+1}}^*$. They proved that the DSMP in $\mathbb{Z}_{n^{s+1}}^*$ is hard if and only if the DSMP is hard in $\mathbb{Z}_{n^2}^*$.

Elliptic curve generalisations Naccache-Stern has an obvious generalisation to elliptic curves. Choose v_1, v_2 prime and u_1, u_2 B-smooth satisfying $\gcd(u_1, u_2) = 1$ and $\gcd(u_1u_2, v_1v_2) = 1$. Use complex multiplication techniques to find primes p_1 and p_2 along with elliptic curves E_1 and E_2 such that

$$\#E_1(\mathbb{F}_{p_1}) = v_1u_1 \quad \text{and} \quad \#E_2(\mathbb{F}_{p_2}) = v_2u_2.$$

The Chinese remainder theorem is then used to find a curve E defined over $\mathbb{Z}_n$ such that $E(\mathbb{Z}_n) \simeq E_1(\mathbb{F}_{p_1}) \times E_2(\mathbb{F}_{p_2})$. Note that one could probably make sure that the endomorphism rings of E_1 and E_2 do not belong to the same quadratic imaginary field, so there is no complex multiplication curve defined over a number field that reduces to the curve E modulo n.

Hasse's theorem says that $\#E_i(\mathbb{F}_{p_i}) = p_i + 1 - t_i$, where t_i, the trace of the Frobenius endomorphism, satisfies $-2\sqrt{p_i} \leq t_i \leq 2\sqrt{p_i}$. The trace is distributed roughly uniformly in this range. We get

$$\begin{aligned}
\#E(\mathbb{Z}_n) &= \#E_1(\mathbb{F}_{p_1})\#E_2(\mathbb{F}_{p_2}) = v_1v_2u_1u_2 \\
&= (p_1 + 1 - t_1)(p_2 + 1 - t_2) \\
&= n - (p_1t_1 + p_2t_2) + (p_1 + p_2) - (t_1 + t_2) + 1 \\
&= n + O(n^{3/4}).
\end{aligned}$$

To break the cryptosystem we need to recover v_1v_2. Using a Baby-step Giant-step method and the knowledge of u_1u_2, it can be recovered using

$O(n^{3/8}/\sqrt{u_1 u_2})$ operations. For optimal security, one would balance this against the difficulty of factoring n.

The advantage of an elliptic curve variant is that the public modulus n no longer has a special form. Presumably, it would therefore be more resistant to factoring. If only the x-coordinate is transmitted, the bandwidth is also slightly higher than for Naccache-Stern. The main disadvantage is that each elliptic curve operation requires several computations in $\mathbb{Z}_n$, so the system would be less efficient. Also, the public key would have to include the elliptic curve parameters, and so be larger.

Galbraith [6] gave an elliptic curve variant of Paillier's system (as well as Damgård and Jurik's generalisation). We note that Okamoto-Uchiyama can also be generalised to elliptic curves using the same methods. Considering $E/\mathbb{Z}_{p^2}$, then

$$E(\mathbb{Z}_{p^2}) \simeq E(\mathbb{F}_p) \times \mathbb{Z}_p,$$

where the $\mathbb{Z}_p$-part is the formal group. The group operation can be computed using the standard formulae, since points will almost never have non-invertible x-coordinates.

5 Security

5.1 Semantic security

Semantic security essentially says that no information about the message can be recovered from the ciphertext, except possibly the length of the message. Broadly speaking, there are three types of attacks: chosen plaintext attacks and adaptive/non-adaptive chosen ciphertext attacks. All of the systems described in Section 4 have been proven semantically secure against chosen plaintext attacks under various assumptions on the groups involved [7, 8, 9, 10].

We will show that the generic system described in Section 2 is semantically secure under the assumption that the DSMP is hard. We say that the *DSMP is hard* for a given G and subgroup K if there does not exists a probabilistic polynomial-time algorithm that decides this problem with probability significantly better than an algorithm that tosses a coin to decide its answer.

Definition. A public-key cryptosystem has *encryptions indistinguishable from random noise* if for any message m and an encryption c of either m or of a message chosen uniformly at random, no probabilistic polynomial time algorithm can decide if c is an encryption of m or not with a significant advantage.

Remark. This is equivalent to semantic security [1].

Theorem 1. *The cryptosystem described in Section 2 has encryptions indistinguishable from random noise if and only if the DSMP is hard for the group G and the subgroup K.*

Proof. First of all, we note that all encryptions of 0 are elements of K. Second, if c is an encryption of m, then $c' = cE(-m, r)$ is an encryption of 0. So the homomorphic property allows any question about a message m to be turned into a question about the message 0, and vice versa.

It is then clear that any algorithm that can decide membership of K can be used to distinguish encryptions of 0, and any algorithm that can distinguish encryptions of m can decide membership of K. □

5.2 Digit security

The basic ideas and techniques in this section are from [2], but our results and proofs are somewhat different so we include them.

We shall prove that the least significant digit of a message constitutes a *hard core predicate* for the encryption system. That is, if one can compute the least significant digit of the message efficiently given only the ciphertext, then one can recover the entire message efficiently and hence invert the encryption function.

We denote the least significant k-digit of a ciphertext c by $lsd(c)$, that is $lsd(c) = D(c) \bmod k$. (Recall that we defined the decryption function result $D(c)$ to be an integer.) Obviously, when $k = 2$ this reduces to questions about bit security.

First we show that any oracle for lsd can be used to recover the message.

Lemma 2. *Suppose $|H|$ is odd, and that s is a publicly known integer such that $ks \equiv 1 \pmod{|H|}$. Given an oracle A'' that computes lsd, the message encoded in a ciphertext can be recovered using $\lceil \log_2 |H| \rceil$ calls to A''.*

Proof. Suppose the message encoded in c is $m = \sum_{i=0}^{l-1} m_i k^i$, so that $lsd(c) = A''(c) = m_0$. Let $m' = (m - m_0)/k$. Then for some integer r we have

$$c = g_K^r g^m = g_K^r g^{km'} g^{m_0}.$$

We get

$$(cg^{-m_0})^s = g_K^{rs}(g^{ks})^{m'} = g_K^{r'} g^{m'}.$$

This is a new ciphertext encoding m', and by induction we can recover the entire message in l steps. □

Now we need to show that any oracle with a significant advantage can be turned into a reliable oracle in probabilistic polynomial time. Let $p_i = \mathrm{Prob}[A'(c) \equiv lsd(c) + i \pmod{k}]$ for $i = 0, \ldots, k-1$. We say that an algorithm A' computes lsd with advantage ϵ if for any message m and any ciphertext c encoding m,

$$p_0 \geq \epsilon + \max\{p_1, p_2, \ldots, p_{k-1}\}. \tag{1}$$

Remark. What does "reliable" mean? The algorithm in Lemma 2 requires $\log_k |H|$ invocations of the algorithm A''. If one requires that the algorithm fail with probability at most α, the algorithm A'' can fail with probability roughly $\alpha / \log_k |H|$.

Lemma 3. *Let $(X_0, X_1, \ldots, X_{k-1})$ be a multinomial random variable with parameters $(p_0, p_1, \ldots, p_{k-1})$ such that for some $\epsilon > 0$, $p_0 \geq p_i + \epsilon$ for $1 \leq i < k$. After l experiments*

$$\mathrm{Prob}[X_0 = \max\{X_0, \ldots, X_{k-1}\}] \geq 1 - \frac{k^3}{p_0^2 \epsilon^3} \frac{1}{l} + O((1/l)^2).$$

We have been unable to find a reference for this result, so we include a proof in the appendix.

Lemma 4. *Let A' be an oracle that computes lsd with advantage ϵ. Suppose two independent invocations of A' will return statistically independent answers. Then there is a algorithm A'' that computes lsd with probability $1 - \alpha/l + O((1/l)^2)$ using l invocations of A', where $\alpha < O(1/\epsilon^8)$.*

Proof. The algorithm A'' will just invoke A' l times, and return the most frequent answer. We may as well suppose that 0 is the correct answer.

Let X_i count the number of times A' returns the value i, $0 \leq i < k$. It is clear that since each invocation of A' is independent, $(X_0, X_1, \ldots, X_{k-1})$ will be a multinomially distributed random variable with parameters $(p_0, p_1, \ldots, p_{k-1})$.

Since A' has advantage ϵ, $p_0 \geq p_i + \epsilon$, $i = 1, \ldots, k-1$. If k is smaller than $O(1/\epsilon)$, the result follows from Lemma 3.

So suppose $k > 4/\epsilon$. For a partition $\{S_1, S_2, \ldots, S_{k'-1}\}$ of $\{1, 2, \ldots, k-1\}$, let $Z_j = \sum_{i \in S_j} X_i$ and $p'_j = \sum_{i \in S_j} p_i$. Let $Z_0 = X_0$ and $p'_0 = p_0$. It is possible to find a partition such that there is at most one $p'_i < \epsilon/4$ and

$$p'_0 \geq p'_i + \frac{\epsilon}{2}, \qquad i = 1, \ldots, k'-1.$$

Now $(Z_0, Z_1, \ldots, Z_{k'-1})$ is a multinomially distributed random variable with parameters $(p'_0, p'_1, \ldots, p'_{k'-1})$, and k' will be at most $4/\epsilon$.

It is clear that

$$\mathrm{Prob}[X_0 = \max\{X_0, \ldots, X_{k-1}\}] \geq \mathrm{Prob}[Z_0 = \max\{Z_0, \ldots, Z_{k'-1}\}].$$

So the result again follows from Lemma 3. □

Remark. The bounds in Lemma 3 and Lemma 4 are not very good. With more knowledge about the exact distribution, much better bounds could probably be found.

Suppose we have an oracle A that returns lsd with an advantage ϵ. We cannot assume that repeated invocations of the oracle will be independent, even if we modify the cloak of the message. That is, the probability $\text{Prob}[A(c) \neq A(cg_K^r)]$ can be negligible.

Lemma 5. *Let A be an oracle that on input c returns $lsd(c)$ with advantage ϵ. If c is an encryption of $m_0 \leq m < m_0 + \lfloor |H|\epsilon/3 \rfloor$, there is a probabilistic polynomial-time algorithm A' that on input c computes $lsd(c)$ with advantage $\epsilon/3$ using a single invocation of A. Repeated invocations of A' are independent.*

Proof. The homomorphic property allows us to modify the ciphertext such that the message changes predictably. If c encodes a general message m, then $cg_K^{r'}g^{m'}$ encodes the message $m + m' \bmod |H|$, that is

$$D(cg_K^{r'}g^{m'}) = \begin{cases} m + m' & m + m' < |H|, \\ m + m' - |H| & m + m' \geq |H|. \end{cases}$$

The problem is that since we do not know m, we do not know when we get $lsd(c)$ or $lsd(c) - |H| \bmod k$.

So assume that $m_0 \leq m < m_0 + \lfloor |H|\epsilon/3 \rfloor$. We compute a ciphertext $c' = cg_K^{r'}g^{m'-m_0}$, which is an encryption of $m + m' - m_0$ with probability at least $1 - \epsilon/3$, and an encryption of $m + m' - m_0 - |H|$ with probability at most $\epsilon/3$. Applying the oracle A to the modified ciphertext, we get an approximation of $lsd(c) - |H| \bmod k$ at most $\epsilon/3$ times, so the advantage is reduced to $\epsilon/3$ in the worst case.

Repeated invocations of this algorithm are independent if the coin tosses that produce r' and m' are independent. □

Theorem 6. *Suppose* $\gcd(|H|, k) = 1$*, and that s is a publicly known integer such that $ks \equiv 1 \pmod{|H|}$. Then lsd is a hard core predicate for the subgroup discrete logarithm problem.*

Proof. Suppose A is an oracle that returns lsd with a significant advantage ϵ, as defined in (1).

First we note that A can be used to decide if a ciphertext encodes zero. This is obvious, since A clearly contradicts semantic security, which was equivalent to deciding if a ciphertext is an encryption of 0 or of a random message.

The strategy is then to divide the total set of messages into a small number of subsets. For each subset, we produce a candidate message that will be correct if the real message is in the subset. Then we simply check which of the candidate messages is the correct one.

Let $\Delta = \lfloor |H|\epsilon/3 \rceil$ and for $i = 0, \dots, \lfloor 3/\epsilon \rfloor$, set

$$S_i = \{i\Delta, (i+1)\Delta - 1\}.$$

Under the assumption that c encodes a message $m \in S_i$, Lemma 5 says that the algorithm A can be turned into a repeatable algorithm A' that computes lsd with advantage $\epsilon/3$. By Lemma 4 this becomes a reliable algorithm A'' (given a proper choice of l), and by Lemma 2 we can recover m. Note that if $m \in S_i$, then $\lfloor m/k \rfloor \in S_{\lfloor i/k \rfloor}$.

Of course, if $m \notin S_i$, the answer will with high probability be incorrect. But since ϵ is significant, $1/\epsilon$ is polynomially bounded, so the number of candidate messages we need to check is polynomially bounded. $\square$

Remark. This theorem also applies to Okamoto-Uchiyama, even though the exact group order is unknown. Any number that is an inverse of k modulo $n = p^2 q$ is also an inverse of k modulo p, and any inverse of k will do in the theorem.

Remark. Let $l = \lfloor \log_k |H| \rfloor$. As in [2], we can show that if there is an algorithm for computing the $l-j$th k-digit of a message with some significant advantage, given only the ciphertext, then any message with less than j digits can be computed.

The basic idea is that any message with only j digits (that is, $D(c) < k^j$) can safely be multiplied with k^{l-j}, to put the least significant digit into the $l-j$th position. This allows us to compute the least significant digit with a significant advantage. The above theorem then tells us how to recover all the j digits of the message.

5.3 Chosen ciphertext attacks

The homomorphic property ensures that the cryptosystems cannot be secure against adaptive chosen ciphertext attacks, since given the encryption of c, the attacker can modify the ciphertext either in a way that does not modify the message, or modifies the message in a predictable way.

The non-adaptive chosen ciphertext attack from [5] can be used against all of the systems in Section 4 based on $\mathbb{Z}_n^*$. Note that it is not a polynomial-time attack, but it can be significantly more efficient than trying to factor the modulus. (Of course, the Okamoto-Uchiyama system falls trivially against a chosen ciphertext attack, because having an oracle that decrypts chosen ciphertexts allows one to factor the modulus.)

The game played in a non-adaptive chosen ciphertext attack is as follows: First, the attacker is given the decryption of a number of ciphertexts chosen by him. He then produces a message m and is given an encryption c either of the message m or of a message chosen uniformly at random.

Consider the systems using subgroups of $\mathbb{Z}_n^*$. Let $B = \{p_1, \dots, p_k\}$ be a set of small primes (possibly including -1). The attacker first obtains the decryptions of each p_i, say l_i. Note that these suspiciously small ciphertexts can easily be camouflaged, using the homomorphic property. (This is essentially the first step of an index-calculus type attack. The chosen ciphertext part gives us the logarithms of the primes for free.)

Now the attacker produces a message m and gets a ciphertext c. He produces random encryptions of known messages $c_j = E(m_j, r_j)$ and tries to factor cc_j as

$$cc_j = \prod_{i=1}^{k} p_i^{\alpha_i}.$$

Once he finds such a factorisation, he knows that

$$m' + m_j \equiv D(cc_j) \equiv D(\prod_{i=1}^{k} p_i^{\alpha_i}) \equiv \sum_{i=1}^{k} \alpha_i D(p_i) \equiv \sum_{i=1}^{k} \alpha_i l_i \pmod{|H|},$$

where m' is the message that was encrypted. Now it is easy to see if $m = m'$.

The attack can also be directed against the Paillier scheme, but we need to be slightly more clever. Again, we suppose that we know the decryptions of a set of primes B, and that we modify the ciphertext by some random encryption $c' = E(r', m')$. Now, since $cc' \in \mathbb{Z}_{n^2}$, we can find a representative on the form $cc' = x_0 + x_1 n$ with $x_0, x_1 \in \{-(n-1)/2, \ldots, (n-3)/2\}$. Suppose we can find a factorisation

$$x_0 = \prod_i p_i^{\alpha_i}.$$

Set $c'' = \prod_i p_i^{-\alpha_i}$. Then

$$cc'c'' \equiv x_0 \prod_i p_i^{-\alpha_i} \equiv 1 \pmod{n}.$$

So we have a ciphertext which is in H. It is trivial to decrypt. Since we know the messages encoded by c' and c'', we can recover the message encoded by c. This attack also applies to the case of $\mathbb{Z}^*_{n^s}$.

It is worth noting that this attack does not work against the elliptic curve variants.

6 A modification

The Naccache-Stern system is not usable in a threshold decryption system, since knowledge of any element in H allows anyone to factor the modulus. We describe a small modification to the system that will allow elements of H to be public.

Let B be a small integer and let u be a B-smooth integer. Let v_1 and v_2 be prime integers such that $p_1 = 2v_1u + 1$ and $p_2 = 2v_2u + 1$ are primes and $\gcd(v_1v_2, u) = 1$. Let $n = p_1p_2$.

Now K is the subgroup of $\mathbb{Z}_n^*$ of order $v_1 v_2$. Let g be an element of order u such that the images of g in $\mathbb{Z}_{p_1}^*$ and $\mathbb{Z}_{p_2}^*$ both have order u. Then H is a subgroup of order u generated by g and $G = HK$. The public key is (n, g). We let $\sigma'(r) = r^2$.

It is clear that knowledge of the element g will not reveal anything about the factorisation of n through simple group operations.

In order to solve the DSMP using the knowledge that u^2 divides $\phi(n)$, we need to be able to determine the order of an element in $\mathbb{Z}_n^*$. This reduces to recovering $v_1 v_2$.

Note that since

$$n = (2v_1 u + 1)(2v_2 u + 1) = 4v_1 v_2 u^2 + 2(v_1 + v_2)u + 1$$

we get that

$$\frac{n-1}{4u^2} = v_1 v_2 + \frac{v_1 + v_2}{2u}.$$

This means that the order $v_1 v_2$ is close to $(n-1)/4u^2$. Using a Baby-step Giant-step type algorithm will require

$$O(\sqrt{(v_1 + v_2)/u}) = O(\sqrt{\sqrt{n}/u^2}) = O(n^{1/4}/u)$$

operations.

Naccache and Stern [8], under the assumption that u_1 and u_2 are known, describe an attack on their cryptosystem of complexity $O(\sqrt{n}/(u_1 u_2)^2)$. For a given security parameter τ, we get for our modification that

$$\frac{1}{4}\log_2 n - \log_2 u = \tau \quad \Leftrightarrow \quad \log_2 u = \frac{1}{4}\log_2 n - \tau,$$

while for the Naccache-Stern cryptosystem we get

$$\frac{1}{2}\log_2 n - 2\log_2 u_1 u_2 = \tau \quad \Leftrightarrow \quad \log_2 u_1 u_2 = \frac{1}{4}\log_2 n - \frac{1}{2}\tau.$$

This means, somewhat surprisingly, that bandwidth is not halved in this variant, as $\log_2 n$ increases much faster than $\log_2 \tau$. The reason is that knowledge of u_1 and u_2 leaks some knowledge of v_1 and v_2, while no knowledge leaks when $u_1 = u_2$.

As the original Naccache-Stern system, this variant extends trivially to elliptic curves. Compared to the elliptic curve variant of Naccache-Stern, bandwidth is really halved.

Remark. This trick can be used for other purposes too. For example, we can speed up a variant of the Paillier scheme. In the original, the cloak K is the part of $\mathbb{Z}_{n^2}^*$ isomorphic to $\mathbb{Z}_n^*$. The variant uses a subgroup K' of K, both to speed up encryption and decryption. If a modulus as above is used, the size of the subgroup K' can be halved, roughly doubling the encryption speed.

Of course, for such use u would be unknown, and should not be smooth.

7 A combination

It is easy to see that we can combine certain of the cryptosystems we have described. The idea is that if the cloak K is very large, some of it could be used to carry information. As an example, we can easily combine the Naccache-Stern and Paillier systems, to achieve a slightly higher bandwidth, at the expense of a larger public key and more decryption work. With $n = (2v_1u_1 + 1)(2v_2u_2 + 1)$, we would have H be the subgroup of $\mathbb{Z}^*_{n^2}$ of order u_1u_2n, and K would be the subgroup of order v_1v_2.

To optimise encryption, one would publish a generator g only for the subgroup of order u_1u_2. Note that $H \simeq \mathbb{Z}_{u_1u_2} \times \mathbb{Z}_n$, so we divide a message m into m_1 and m_2 such that $m \equiv m_1 \pmod{u_1u_2}$ and $m \equiv m_2 \pmod{n}$. Then we encrypt a message m as $g_C^r g^{m_1}(1 + m_2n)$.

To decrypt, one can recover m_2 as in Paillier's scheme, then reduce the ciphertext modulo n and recover m_1 as in the Naccache-Stern scheme.

As usual, these concepts have natural generalisations to elliptic curves.

Theorem 7. *Let v_1, v_2, u_1, u_2, n be as above. Let K, H_1 and H_2 be the subgroups of $\mathbb{Z}^*_{n^2}$ of order v_1v_2, u_1u_2 and n, respectively. Let $G = KH_1H_2$. Likewise, let K' and H'_1 be the subgroups of $\mathbb{Z}^*_n$ of order v_1v_2 and u_1u_2, respectively, and let $G' = K'H'_1$.*

The DSMP (G, K) is hard if and only if both the DSMP (G', K') and the DSMP (G, KH_1) are hard.

Proof. Reduction modulo n gives a map from G to G'. The kernel is obviously H_2, and elements in G of order v_1v_2 go to elements in G' of order v_1v_2. If $c' \in G'$ is the image of $c \in G$, then $c' \in K'$ if and only if $c \in KH_2$.

An element of H_1H_2 chosen at random will be in H_1 or H_2 with negligible probability. So any algorithm that answers either the DSMP (G', K') or the DSMP (G, KH_1) with a significant advantage can solve the DSMP (G, K) with only negligible worse advantage.

We now prove the converse. First we note that we can turn an instance c' of the DSMP (G', K') into an instance of the DSMP (G, KH_2), simply by choosing a representative x and then choose c to be the residue class of x modulo n^2. It is clear that $c \in KH_2$ if and only if $c' \in K$.

An instance c of (G, KH_2) can of course be turned into an instance of the DSMP (G, K), by computing c^n. Likewise, an instance c of the DSMP (G, KH_1) can be turned into an instance of the DSMP (G, K) by computing $c^{u_1u_2}$. But the distribution of these instances will be significantly different from instances of the DSMP (G, K) drawn uniformly at random.

Now suppose A is an algorithm that can solve the DSMP (G, K), that is, it can distinguish elements of K from elements of $G \setminus K$ with significant probability. Suppose it also has negligible advantage when trying to distinguish elements of K from elements of KH_1. Then the algorithm must have a significant advantage when distinguishing elements of KH_1 from elements of KH_1H_2, so it can solve the DSMP (G, KH_1).

Likewise, if A has negligible advantage when trying to distinguish elements of K from elements of KH_2, it can solve the DSMP (G, KH_2).

And finally, if A has non-negligible probability of solving either (KH_1, K) or (KH_2, K), then obviously we can solve either (G', K') or (G, KH_1). $\square$

Remark. Since the modulus for the Naccache-Stern system is somewhat special, it is clear that the DSMP for the combined system could be easy, even though the DSMP for the Paillier system with a more general modulus is hard.

8 Concluding remarks

We have described a general framework for a certain type of homomorphic cryptosystems, generalising most of the proofs to this setting. The advantage is to simplify descriptions and security proofs. Of course, the security still depends on the concrete DSMP. We have also described two new cryptosystems, with either new properties or better bandwidth.

All of the cryptosystems described are essentially based on the factoring problem. If factoring is not hard, then the above systems can all be broken. It would be very interesting to find a group satisfying the requirements of Section 2, but where the difficulty of the SDLP and DSMP did not depend on the difficulty of factoring.

A Proof of Lemma 3

Let Y_i represent the event $|X_i/l - p_i| \leq \epsilon/k$, $1 \leq i < k$. Then

$$\mathrm{Prob}[Y_i] = 1 - \frac{p_i(1-p_i)}{l\epsilon^2/k^2}.$$

Let Y_0 represent the event $|X_0/l - p_0| \leq (k-1)\epsilon/k$. We have

$$\begin{aligned}\mathrm{Prob}[X_0 = \max\{X_i\}] &\geq \mathrm{Prob}[Y_0 \wedge Y_1 \wedge \cdots \wedge Y_{k-1}] \\ &= \mathrm{Prob}[Y_0 | Y_1 \wedge \cdots \wedge Y_{k-1}] \cdots \mathrm{Prob}[Y_{k-2}|Y_{k-1}]\,\mathrm{Prob}[Y_{k-1}].\end{aligned}$$

If we assume $Y_1 \wedge \cdots \wedge Y_{k-1}$ and l experiments, then

$$\begin{aligned}|X_0 - lp_0| &= |l - X_1 - \cdots - X_{k-1} - l(1 - p_1 - \cdots - p_{k-1})| \\ &= |X_1 - lp_1 + X_2 - lp_2 + \cdots + X_{k-1} - lp_{k-1}| \\ &\leq |X_1 - lp_1| + \cdots + |X_{k-1} - lp_{k-1}| \\ &\leq \frac{\epsilon}{k} + \ldots \frac{\epsilon}{k} = \frac{k-1}{k}\epsilon.\end{aligned}$$

It follows that $\text{Prob}[Y_0|Y_1 \wedge \cdots \wedge Y_{k-1}] = 1$.

Now we consider $\text{Prob}[Y_i|Y_{i+1} \wedge \cdots \wedge Y_{k-1}]$. The marginal distribution of X_i has parameter $p_i/(1-\sum_{j=i+1}^{k-1} p_j)$, and there are a total of $l' = l - \sum_{j=i+1}^{k-1} X_j$ experiments.

Under the condition $Y_{i+1} \wedge \cdots \wedge Y_{k-1}$, we get that

$$l - \sum_{j=i+1}^{k-1} (lp_j + l\epsilon/k) \leq l' \leq l - \sum_{j=i+1}^{k-1} (lp_j - l\epsilon/k).$$

Then

$$\begin{aligned}
\text{Prob}[Y_i|Y_{i+1} \wedge \cdots \wedge Y_{k-1}] &= 1 - \frac{\frac{p_i}{1-\sum_{j=i+1}^{k-1} p_j}\left(1 - \frac{p_i}{1-\sum_{j=i+1}^{k-1} p_j}\right)}{l'\epsilon^2/k^2} \\
&= 1 - \frac{k^2 p_i (1 - \sum_{j=i}^{k-1} p_j)}{l'(1-\sum_{j=i+1}^{k-1} p_j)\epsilon^2} \\
&\geq 1 - \frac{k^2 p_i (1 - \sum_{j=i}^{k-1} p_j)}{l(1-\sum_{j=i+1}^{k-1}(p_j + \epsilon/k))(1-\sum_{j=i+1}^{k} p_j)\epsilon^2} \\
&\geq 1 - \frac{k^2 p_i}{l(p_0 - (k-1)\epsilon/k)p_0^2\epsilon^2} \\
&\geq 1 - \frac{k^3 p_i}{l p_0^2 \epsilon^3}.
\end{aligned}$$

We get

$$\begin{aligned}
\text{Prob}[Y_0 \wedge \cdots \wedge Y_{k-1}] &= \prod_{i=1}^{k-1} \text{Prob}[Y_1|Y_{i+1} \wedge \cdots \wedge Y_{k-1}] \\
&\geq \pi_{i=1}^{k-1}(1 - \frac{k^3 p_i}{p_0^2\epsilon^3}\frac{1}{l}) \\
&= 1 - \frac{k^r(1-p_0)}{p_0^2\epsilon^3} + O((1/l)^2).
\end{aligned}$$

The result follows.

References

[1] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *FOCS 1997*, pages 394–403, 1997.

[2] Dario Catalano, Rosario Gennaro, and Nich Howgrave-Graham. The bit security of Paillier's encryption scheme and its applications. In

B. Pfitzmann, editor, *Proceedings of Eurocrypt 2001*, volume 2045 of *LNCS*, pages 229–243. Springer-Verlag, 2001.

[3] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *Proceedings of Public Key Cryptography 2001*, volume 1992 of *LNCS*, pages 119–136. Springer Verlag, 2001.

[4] Ivan Damgård, Jens Groth, and Gorm Salomonsen. The theory and implementation of an electronic voting system. In D. Gritzalis, editor, *Secure Electronic Voting*. Kluwer Academic Publishers, 2002.

[5] Y. Desmedt and A. M. Odlyzko. A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In *Advances in Cryptology – Crypto '85*, number 218 in Lecture Notes in Computer Science, pages 516–521. Springer-Verlag, 1986.

[6] Steven D. Galbraith. Elliptic curve Paillier schemes. *Journal of Cryptology*, 15(2):129–138, 2002.

[7] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, April 1984.

[8] D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proceedings of Eurocrypt '98*, volume 1403 of *LNCS*, pages 308–318. Springer Verlag, 1998.

[9] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Proceedings of Eurocrypt '98*, volume 1403 of *LNCS*, pages 308–318. Springer Verlag, 1998.

[10] P. Paillier. Public-key cryptosystems based on composite degree residue classes. In *Proceedings of Eurocrypt '99*, volume 1592 of *LNCS*, pages 223–238. Springer-Verlag, 1999.

[11] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.

[12] Victor Shoup. Practical threshold signatures. *Lecture Notes in Computer Science*, 1807:207–220, 2000.

[13] Akihiro Yamamura and Taiichi Saito. Private information retrieval based on the subgroup membership problem. In V. Varadharajan and Y. Mu, editors, *Proceedings of ACISP 2001*, volume 2119 of *LNCS*, pages 206–220. Springer-Verlag, 2001.