Cryptanalysis of a Provable Secure Additive and Multiplicative Privacy Homomorphism

Feng Bao
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
baofeng@i2r.a-star.edu.sg

Abstract

Additive and multiplicative privacy homomorphisms (PHs) are encryption transformations that map the addition and multiplication on plaintext to the addition and multiplication on ciphertext. PHs have important applications in encrypted data processing. However, it has been shown that no PH can be secure against chosen-plaintext attack if addition is the operation that can be mapped from plaintext to ciphertext. Hence the best security an additive and multiplicative PH can achieve is to be secure against known-plaintext attack, while for many applications of PHs such security is good enough. A new additive and multiplicative PH is proposed in [7], and is claimed to be provably secure against known-plaintext attack. In this paper we break the scheme with known-plaintext attack and we need only a small number of known plaintext-ciphertext pairs. Our result indicates that the security proof presented in [7] must be lacking rigour.

1 Introduction

The concept of PH (privacy homomorphism) was first proposed by Rivest, Adlemann and Dertouzos in [11]. A PH is an encryption transformation that maps some operations on plaintext to some operations on ciphertext. An additive and multiplicative PH keeps addition and multiplication from plaintext to ciphertext. If we denote a PH by $E_k(.)$ and the corresponding decryption by $D_k(.)$, E_k is an additive and multiplicative PH if $D_k(E_k(m_1) + E_k(m_2)) = m_1 + m_2$ and $D_k(E_k(m_1) \cdot E_k(m_2)) = m_1 \cdot m_2$. Please note that it is not accurate to write $E_k(m_1) + E_k(m_2) = E_k(m_1 + m_2)$ and $E_k(m_1) \cdot E_k(m_2) = E_k(m_1 \cdot m_2)$ since E_k may be a randomized encryption.

PHs are useful in encrypted data processing, such as secure computation of statistics [2], [6]. A typical scenario of the application of PH is as follows. A client encrypts some sensitive data and stores the ciphertexts to a remote database that provides storage service. When the client needs to compute some statistics of the stored data, he may neither like to let the database decrypt the ciphertexts and do the statistics computation on the plaintexts, nor like to ask the database to transfer all the encrypted data to him. The best way is to ask the database conduct the computation over the encrypted data and send the results back. He then decrypts the results to get the statitics on plaintexts. PHs can also be applied in mobile agent security [12], e-gambling [7], and some other applications [1], [8].

A recent work [4] on privacy-preserving statistics utilizes the homomorphic encryption schemes in [10] and [9], which are PHs that map the addition on plaintext to the multiplication on ciphertext. The schemes of [10] and [9] are public encryption systems and hence are

quite expensive in computation. And only one operation on plaintext can be mapped to one operation on ciphertext, which limits the application of the schemes in encrypted data processing. Additive and multiplicative PHs are more flexible and effective in privacy-preserving statistics.

The first additive and multiplicative PH proposed in [11] was broken with known-plaintext attack in [3]. It has been shown in [1] that no additive PH can be secure against chosen-plaintext attack. Hence the best security an additive and multiplicative PH can achieve is to be secure against known-plaintext attack. For many applications, such as the secure statistics scenario described above, the security against known-plaintext attack is acceptable. The client will not encrypt the data chosen by an attacker.

An additive and multiplicative PH was proposed in [5] and it stays secure against known-plaintext attack so far. However, the scheme of [5] has a low efficiency in multiplication of ciphertexts. The main problem is that it is done in integer instead of modular integer. Hence the products of the ciphertexts consist of the bigger and bigger integers, so that the multiplication of ciphertexts is too heavy in computation.

In [7], a new additive and multiplicative PH is proposed and it is much improved in multiplication efficiency. All the computations are conducted in $\mathbf{Z}_m$ for a public m. Moreover, the scheme is claimed to be provably secure against known-plaintext attack with the proof presented in the same paper.

In this paper, we show that the proof must be wrong since we can break the scheme with known-plaintext attack. We only need a small number of plaintext-ciphertext pairs to recover the secret key. There is no special requirement on the known pairs. The rest of the paper is organized as follows. In Section 2, we introduce the PH scheme of [7]. In Section 3, we present our attack. In Section 4, we give the performance analysis of the attack. Section 5 concludes the paper.

2 Description of the Privacy Homomorphism

We follow the same denotations adopted in [7].

Public Parameters

- d – An integer larger than 2, $d > 2$.
- m – A large integer ($m > 10^{200}$). m should have many small divisors and at the same time there should be many integers less than m that are co-prime with m.

d and m can be either shared by a group of users or owned by one user. In the latter case, each user must generate his own d, m and publish them. It is also possible that a group of users share m while choosing their own d. The attack presented in this paper is irrelevant with whether the public parameters are shared or not.

Secret Key

- r – An integer co-prime with m.
- m' – A factor of m.

Encryption

For a plaintext $a \in \mathbf{Z}_{m'}$ and secret key $k = (r, m')$,

$$E_k(a) = (a_1 r \bmod m,\ a_2 r^2 \bmod m,\ \cdots,\ a_d r^d \bmod m)$$

where $a_1,\ a_2,\ \cdots,\ a_d$ is a random split of a such that $a = \sum_{j=1}^{d} a_j \bmod m'$ for $a_j \in \mathbf{Z}_m$.

The encryption is a transformation from unknown domain $\mathbf{Z}_{m'}$ to $\{\mathbf{Z}_m\}^d$. Unknown encryption domain is also adopted in the schemes of [10] and [9].

Decryption

$$D_k((c_1,\ c_2,\ \cdots,\ c_d)) = \sum_{j=1}^{d} c_j/r^j \bmod m'$$

Addition

They are done componentwise, i.e., between terms of same degree.

Multiplication

It works like in the case of polynomials: all terms are cross-multiplied in $\mathbf{Z}_m$.with a d_1-degree term by a d_2-degree term yielding a $(d_1 + d_2)$-degree term; finally, terms having the same degree are added up.

Scalar Multiplication

Let $c \in \mathbf{Z}_m$. Define $c(c_1,\ c_2,\ \cdots,\ c_d) = (cc_1,\ cc_2,\ \cdots,\ cc_d)$. We have $D_k(cE_k(a)) = (ca \bmod m')$.

Before presenting our attack, we would like to briefly review the scheme in [5] so that it is easy to see how the efficiency is improved in [7].

Scheme of [5] Plaintext domain is $\mathbf{Z}_m$ where $m = pq$. m is public but p, q are secret. r_p and r_q are the other two secret keys. Encryption of $a \in \mathbf{Z}_m$ is $E_k(a) = ([a_1 r_p \bmod p, a_1 r_q \bmod q], [a_2 r_p^2 \bmod p, a_2 r_q^2 \bmod q], \cdots, [a_d r_p^d \bmod p, a_d r_q^d \bmod q])$, where $(a_1, a_2, \cdots, a_d)$ is a random split of a. The addition and multiplication are similar to those of [7] except that they are done in integer instead of $\mathbf{Z}_m$. The difference of [7] from [5] is that the ciphertext domain is known in [7] but unknown in [5]. Therefore the multiplication and addition can only be conducted in integer in [5]. The computational complexity and the memory complexity are improved from $O(n^2)$ in [5] to $O(n)$ in [7] for a product of n ciphertexts.

3 Known-Plaintext Attack of the Scheme

In the scheme of [7], m' is a secret that should not be obtained by others. This is the reason why m is required to have many small divisors. In that case, it is not feasible to guess m' since there are too many possible combinations of m'. However, m is not supposed to be difficult to be factorized. In fact, factorizing m is not difficult since m has many small divisors. The scheme is not claimed to be secure based on the hardness of factorizing m.

Suppose we know the factorization of m, and we have some plaintext-ciphertext pairs. We can quickly figure out whether a prime factor p of m is also a factor of m'. By trying the prime factors of m one by one, we can compute m' and r very efficiently.

Idea of the Attack Before going to the details of the attack, we describe it informally for a quick understanding of the attack. Let $a = a_1 + a_2 + \cdots + a_d \bmod m'$ and $E_k(a) = (c_1, c_2, \cdots, c_d)$ where $c_j = a_j r^j \bmod m$. Let p be a factor of m. If p is also a factor of m', we must have $a = c_1 R + c_2 R^2 + \cdots + c_d R^d \bmod p$ for some R ($R = 1/r \bmod p$). If we have a set of plaintext-ciphertext pairs, the R is same for all of them, i.e., the equation group has

a solution. If p is not a factor of m', $a = c_1R + c_2R^2 + \cdots + c_dR^d \bmod p$ holds with only a probability $1/p$, considering that $(a_1, a_2, \cdots, a_d)$ is a random split of a. Hence when we have many known plaintext-ciphertext pairs, the equation group has no solution. In this way we can try all the prime factors of m and determine m' finally.

Known-Plaintext Attack

Let $m = p_1^{e_1}p_2^{e_2}\cdots p_t^{e_t}$ for primes $p_1, p_2, \cdots, p_t$, and $[b_i, (c_{i1}, c_{i2}, \cdots, c_{id})]$, $i = 1, 2, \cdots, n$, be n known plaintext-ciphertext pairs. We denote $X \bmod p$ by $[X]_p$.

1. First we check whether p_1 is a factor of m'. Define $f_i(x) \in \mathbf{Z}_{p_1}[x]$ for $i = 1, 2, \cdots, n$

$$f_i(x) = [c_{id}]_{p_1}x^d + \cdots + [c_{i1}]_{p_1}x - [b_i]_{p_1}$$

   Try $x = 0, 1, 2, \cdots, p_1 - 1$ and check if there is a solution such that $f_i(x) = 0$ for all $i = 1, 2, \cdots, n$. If yes, we denote the solution by R_1 ($R_1 = [1/r]_{p_1}$ in our setting) and we assume that p_1 is a factor of m'. We will show later that it is unlikely to have two solutions. If there is no solution at all, we assume that p_1 is not a factor of m'. (Note. If p_1 is large, it is not efficient to try $x = 0, 1, 2, \cdots, p_1 - 1$. In that case, we can compute the greatest common divisor $gcd(f_1(x), f_2(x), \cdots, f_n(x))$ by Euclidean Algorithm. If $gcd(f_1(x), f_2(x), \cdots, f_n(x)) = x - R_1$, that R_1 is what we want. Another method is to take $x, x^2, \cdots, x^d$ as d variables and solve the linear equation group. See Section 4.)

2. If p_1 is identified as a factor of m', we need to check whether p_1^2 is a factor of m' too. From 1, we have

$$\sum_{j=1}^{d}[c_{ij}]_{p_1}R_1^j - [b_i]_{p_1} = 0 \bmod p_1 \tag{1}$$

   for $i = 1, 2, \cdots, n$.

   Define $g_i(x) \in \mathbf{Z}_{p_1^2}[x]$ for $i = 1, 2, \cdots, n$

$$g_i(x) = \sum_{j=1}^{d}[c_{ij}]_{p_1^2}x^j - [b_i]_{p_1^2}$$

   Substituting $R_2p_1 + R_1$ into $g_i(x) = 0$ for $i = 1, 2, \cdots, n$, we obtain

$$R_2p_1\sum_{j=1}^{d}jR_1^{j-1}[c_{ij}]_{p_1^2} + \sum_{j=1}^{d}[c_{ij}]_{p_1^2}R_1^j - [b_i]_{p_1^2} = 0 \bmod p_1^2 \tag{2}$$

   Due to (1) we have

$$\sum_{j=1}^{d}[c_{ij}]_{p_1^2}R_1^j - [b_i]_{p_1^2} = 0 \bmod p_1$$

   Hence we obtain n linear equations modulo p_1 for R_2 from (2).

$$R_2\sum_{j=1}^{d}jR_1^{j-1}[c_{ij}]_{p_1^2} + \frac{\sum_{j=1}^{d}[c_{ij}]_{p_1^2}R_1^j - [b_i]_{p_1^2}}{p_1} = 0 \bmod p_1$$

   If the solutions for the n equations are identical, we assume that p_1^2 is a factor of m' and take $[1/r]_{p_1^2} = R_2p_1 + R_1$. Otherwise we assume p_1^2 is not a factor of m'.

3. If p_1^2 is a factor of m', we continue the process similar to 2 until we find the maximum $u \leq e_1$ such that p_1^u is a factor of m'. And we compute $[1/r]_{p_1^u}$.

4. Perform 1, 2 and 3 for $p_2, p_3, \cdots, p_t$. Finally we obtain m'. By Chinese Remainder Theorem, we can also obtain $[1/r]_{m'}$, and therefore, $[r]_{m'}$. m' and $[r]_{m'}$ can be used to decrypt any ciphertext.

4 Performance Analysis of the Attack

Now we look at how large the n should be such that the attack has large successful probability.

Proposition 1. Let p be a prime and $a_{ij} \in_{random} \mathbf{Z}_p$ for $i = 1, 2, \cdots, n$ and $j = 0, 1, 2, \cdots, d$. Denote $f_i(x) = \sum_{j=0}^{d} a_{ij} x^j \in \mathbf{Z}_p[x]$. The probability that there exists a solution such that $f_i(x) = 0$ for $i = 1, 2, \cdots, n$ is smaller than $1/p^{n-1}$.

Proof It is easy to see that

$$Prob\{\text{there exists a solution}\} < \sum_{x=0}^{p-1} Prob\{f_i(x) = 0 | i = 1, 2, \cdots, n\}$$

Proposition 2. Let p be a prime and $a_{ij} \in_{random} \mathbf{Z}_{p^e}$ for $i = 1, 2, \cdots, n$ and $j = 0, 1, 2, \cdots, d$. Denote $f_i(x) = \sum_{j=0}^{d} a_{ij} x^j \in \mathbf{Z}_{p^e}[x]$. The probability that there exists a solution such that $f_i(x) = 0$ for $i = 1, 2, \cdots, n$ is smaller than $1/p^{en-1}$.

Proposition 3. Let p be a prime and $a_{ij} \in_{random} \mathbf{Z}_p$ for $i = 1, 2, \cdots, n$ and $j = 1, 2, \cdots, d$. Denote $f_i(x) = \sum_{j=1}^{d} a_{ij} x^j - \sum_{j=1}^{d} a_{ij} r^j$. Then r is a solution such that $f_i(x) = 0$ for $i = 1, 2, \cdots, n$. The probability that there exist two solutions such that $f_i(x) = 0$ for $i = 1, 2, \cdots, n$ is smaller than $(p-1)/p^n$.

Since the plaintext is randomly split, those $[c_{ij}]_p$ are random from $\mathbf{Z}_p$. If p is a factor of m', we have $[b_i]_p = \sum_{j=1}^{d} [c_{ij}]_p (1/r)^j \bmod p$. The situation is similar to that of Proposition 3. If p is not a factor of m', $[b_i]_p$ can be equally probable to be any element of $\mathbf{Z}_p$, which is the situation of Proposition 1.

Solving Linear Equation Group In the Step 1 of the attack, we need to verify whether the equation group $f_i(x) = 0$, for $i = 1, 2, \cdots, n$, has a solution, and find the solution if a solution exists. For small p, we can just compute $f_i(x)$ for $x = 0, 1, \cdots, p-1$. When p is large, we can solve the linear equation group

$$[c_{id}]_p x_d + \cdots + [c_{i1}]_p x_1 - [b_i]_p = 0, \ \ i = 1, 2, \cdots, n$$

If there is no solution, p is not assumed to be a factor of m'. If there is a solution, check whether $x_j = x_1^j \bmod p$ for $j = 1, 2, \cdots, t$. If yes, take p as a factor of m'. Otherwise, p is not assumed to be a factor of m'.

Proposition 4. Let p be a prime and $c_{ij} \in_{random} \mathbf{Z}_p$ for $i = 1, 2, \cdots, n$ and $j = 1, 2, \cdots, d$, $n > d$. The probability that $rank(v_1, v_2, \cdots, v_n) = d$ is larger than $1 - 1/p^{n-d}$, where $v_i = (c_{i1}, c_{i2}, \cdots, c_{id})$.

Proposition 4 guarantees that the linear equation group above either has one solution or has no solution except for a negligible probability.

Successful Probability of the Attack

Let $m = p_1^{e_1} p_2^{e_2} \cdots p_t^{e_t}$. Suppose we have n known plaintext-ciphertext pairs, the probability that the attack fails is smaller than

$$\frac{e_1}{p_1^{n-1}} + \frac{e_2}{p_2^{n-1}} + \cdots + \frac{e_t}{p_t^{n-1}}$$

Practical Situations Consider $m > 10^{200}$, as recommended in [7]. When $m \approx 10^l$, we have $\sum_{i=1}^{t} e_i \leq 10l/3$. Hence $Prob\{\text{fail}\} << (10l/3)(1/2^{n-1})$ $(p_i \geq 2)$.

	$m \approx 10^{200}$	$m \approx 10^{300}$	$m \approx 10^{400}$	$m \approx 10^{500}$
$n = 15$	0.04	0.06	0.08	0.1
$n = 20$	0.0125	0.019	0.025	0.032
$n = 25$	0.00004	0.00006	0.00008	0.0001

Table 1: The up-bounds of the probabilities the attack fails

Table 1 only gives very loose up-bounds, where we take $1/2$ as the up-bound of each $1/p_i$. In practical situations each e_i should not be large in order to get as many combinations of m' as possible. The ideal situation is $e_i = 1$. In that case most of p_i's are larger than 10. Then we can get much better up-bounds than Table 1.

If all p_i's are larger than 10, we need only $n = 6, 7, 9$ to obtain the same results for the situations of $n = 15, 20, 25$ in Table 1. That is, 10 known plaintext-ciphertext pairs make our attack successful except for a small probability.

5 Conclusion

In this paper we break the PH scheme proposed in [7] by known-plaintext attack, which is claimed to have provable security against known-plaintext attack. The result enforces the opinion expressed in [14] that security proofs also need time to be validated through public discussion. Before the validation, a cryptosystem with security proof is not superior than a scheme without security proof. Sometimes the former may be even more harmful than the latter since it may cause wrong trust. However, validating a security proof is by no means a simple job. Even the widely believed (original) security proof of OAEP was found having gap [13]. Therefore it may be a good practice to try various attacks to a new scheme even if it has a security proof.

It might be an interesting topic for further research to design additive and multiplicative PHs that are secure against known-plaintext attack meanwhile keep the size of the ciphertext fixed (even after addition and multiplication), or to prove that no such PH exists.

References

[1] N. Ahituv, Y. Lapid and S. Neumann, "Processing encrypted data", Communications of the ACM, vol. 20, no. 9, pp. 777-780, Sept 1987.

[2] G. R. Blakley and C. Meadows, "A database encryption scheme which allows the computation of statistics using encrypted data", Proceedings of the IEEE Symposium on Research in Security and Privacy, New York: IEEE CS Press, pp. 116-122, 1985.

[3] E. F. Brickell and Y. Yacobi, "On privacy homomorphisms", Proceedings of Eurocrypt'87, Lecture Notes in Computer Science, Springer-Verlag, pp. 117-125, 1988.

[4] W. Du and M. J. Atallah, "Privacy-preserving cooperative statistical analysis", Proceedings of the 2001 Annual Computer Security Applications Conference (ACSAC), ACM SIGSAC, New Orleans, 2001.

[5] J. Domingo-Ferrer, "A new privacy homomorphism and applications", Information Processing Letters, vol. 60, no. 5, pp. 277-282, Dec 1996.

[6] J. Domingo-Ferrer and R. X. Sanchez del Castillo, "An implementable scheme for secure delegation of statistical data", Proceedings of ICICS'97, Lecture Notes in Computer Science 1334, Springer-Verlag, pp. 445-451, 1997.

[7] J. Domingo-Ferrer, "A provable secure additive and multiplicative privacy homomorphism", Proceedings of ISC'2002, Lecture Notes in Computer Science 2433, Springer-Verlag, pp. 471-483, 2002.

[8] J. Feigenbaum and M. Merritt, "Open questions, talk abstracts, and summary of discussions", DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 2, pp. 1-45, 1991.

[9] T. Okamoto and S. Uchiyama, "An efficient public key cryptosystem", Proceedings of Eurocrypt'98, Lecture Notes in Computer Science, Springer-Verlag, pp. 308-318, 1998.

[10] P. Paillier, "Public key cryptosystems based on composite degree residue classes", Proceedings of Eurocrypt'99, Lecture Notes in Computer Science, Springer-Verlag, pp. 223-238, 1999.

[11] R. L. Rivest, L. Adleman and M. L. Dertouzos, "On data banks and privacy homomorphisms", Foundations of Secure Computation, R. A. DeMillo et al, Eds. New York: Academic Press, pp. 169-179, 1978.

[12] T. Sander and C. F. Tschudin, "Protecting mobile agents against malicious hosts", Proceedings of Mobile Agent Security, Lecture Notes in Computer Science 1419, Springer-Verlag, pp. 44-60, 1998.

[13] V. Shoup, "OAEP reconsideration", Proceedings of Crypto'01, Lecture Notes in Computer Science 2134, Springer-Verlag, pp. 239-259, 2001.

[14] J. Stern, D. Pointcheval, J. Lee and N. Smart, "Flaws in applying proof methodologies to signature schemes", Proceedings of Crypto'02, Lecture Notes in Computer Science 2442, Springer-Verlag, pp. 93-110, 2002.