On the sparse subset sum problem from Gentry-Halevi's implementation of fully homomorphic encryption

Moon Sung Lee

National Institute for Mathematical Sciences, Daejeon, KOREA 151-742
mslee@nims.re.kr

Abstract. In Gentry's fully homoomrphic cryptosystem, a sparse subset sum problem is used and a big set is included in the public key. In the implementation of a variant of Gentry's scheme, to reduce the size of the public key, Gentry and Halevi used a specific form of a sparse subset sum problem with geometric progressions. In this note, we show that their sparse subset sum challenges are rather easy given the aggressive choice of parameters. Our experiment shows that even their large instance of a sparse subset sum problem could be solved within two days with probability of about 44%. A more conservative parameter choice can easily avoid our attack.

Keywords: sparse subset sum, lattice reduction, dimension reduction method, geometric progression, homomorphic encryption

1 Introduction

In 2009, Gentry described a fully homomorphic cryptosystem based on ideal lattices [2]. Following Gentry's framework, several variants and optimizations of Gentry's fully homomorphic encryption were introduced [13, 11, 4, 3, 12]. In Gentry's scheme, he first constructed a "somewhat homomorphic" scheme that supports evaluating low-degree polynomials on the encrypted data, squashing the decryption procedure to a low-degree polynomial that is supported by the scheme. In this procedure, he used a sparse subset sum and assumed that given big set of integers and a target sum, it would be difficult to find a sparse subset that sums up to a target.

Recently, a variant of Gentry's scheme was implemented [4, 3]. In [3], Gentry and Halevi showed several optimizations that allow them to implement all aspects of the scheme, providing concrete parameter sets with Internet challenges which consist of main challenges and underlying sparse subset sum challenges. One of the optimizations used to reduce the public key size in [3] was to construct a specific form of the sparse subset sum. Their sparse subset sum consists of big sets defined as geometric progressions such that sum of elements in each big set equals the secret key.

In this paper, we study this specific form of the sparse subset sum problem (SSSP), showing that it is relatively easy to solve these SSSP challenges due to the specific form and aggressive parameter choice made in the [3]. Our analysis and experiments show that even the large SSSP challenge is solvable within two days with a high probability.

We present two methods with which to attack this problem. The first is a dimension reduction method which was used in previous research [9, 7]. The second is a probabilistic method which is more efficient. Both methods use lattice reduction algorithms with slightly different lattices from those considered in [3].

The rest of the paper is organized as follows. We first review lattices and the sparse subset sum problem in the next section. In Section 3, we review a specific form of a sparse subset sum problem and the lattice-based attack considered in [3] . Our attacks and experimental results are shown in Section 4, after which we discuss the secure parameter briefly in Section 5. We conclude the paper in Section 6.

2 Background

2.1 Lattices

A full-rank n-dimensional *lattice* is a discrete subgroup of $\mathbb{R}^n$, represented as the set of all integer combinations of some *basis* matrix B whose rows are linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ in $\mathbb{R}^n$. This lattice is denoted as $L = \mathcal{L}(B)$. The determinant of a lattice L is denoted as $\det(L) = |\det(B)|$.

Short vectors and Lattice basis reduction. The length of the shortest nonzero vector in a lattice L is denoted $\lambda_1(L)$. Minkowski's theorem says that we have $\lambda_1(L) < \sqrt{n}\det(L)^{1/n}$ for any n-dimensional lattice L. Heuristically, for random lattices we do not expect to find any vectors of length l if $l \ll \det(L)^{1/n}$. However, for $l \gg \det(L)^{1/n}$, we expect exponentially many vectors of length l.

Short vectors can be found using lattice basis reduction algorithms, for example, the LLL [8] and BKZ [10] algorithms. In small dimensions, the first vector of the output basis of the LLL algorithm is often shortest nonzero vector in a lattice.

2.2 Sparse Subset Sum Problem

Given a big set of integers $A = \{a_1, a_2, \ldots, a_n\}$, t and M in $\mathbb{Z}$, a sparse subset sum problem requires a very sparse subset of a big set which sums up to t modulo M. In general, a sparse subset sum problem is considered to be hard, and lattice reduction algorithms do not seem to apply when n is quite large. For efficiency reasons, Gentry and Halevi used geometric progressions in the big set. Details of the construction and lattice-based attack considered in [3] are explained in the next section.

3 The Sparse Subset Sum Construction

In [3], instances of a sparse subset sum problem consist of s big sets $\mathcal{B}_1, \dots, \mathcal{B}_s$, each with S elements in $\mathbb{Z}_d$ such that the sum of the elements in each $\mathcal{B}_k$ equals the secret key w modulo d. The public key includes s big sets, namely $s \cdot S$ elements from $\mathbb{Z}_d$. To reduce the size of this public key, big sets are defined as geometric progressions in $\mathbb{Z}_d$: the k'th big set $\mathcal{B}_k$ consists of the elements $x(k,i) = \langle x_k \cdot R^i \rangle_d$ for $i = 0, 1, \dots, S-1$, where R is some public parameter and $\langle z \rangle_d$ represents a reduction of z modulo d. Thus, there is a single secret index i_k in every big set such that

$$\sum_k x(k, i_k) = \sum_k x_k \cdot R^{i_k} = w \pmod{d}. \tag{1}$$

At this point, this specific form of SSSP requests these secret indexes.

Gentry and Halevi chose the parameter R to avoid lattice-based attacks. More specifically, the following basis is considered to set the parameter R,

$$B = \begin{pmatrix} 1 & & & & & x(1,0) \\ & 1 & & & & x(2,0) \\ & & \ddots & & & \\ & & & 1 & & x(s,0) \\ & & & & 1 & -w \\ & & & & & d \end{pmatrix} = \begin{pmatrix} 1 & & & & & x_1 \\ & 1 & & & & x_2 \\ & & \ddots & & & \\ & & & 1 & & x_s \\ & & & & 1 & -w \\ & & & & & d \end{pmatrix}. \tag{2}$$

Clearly, the lattice $L = \mathcal{L}(B)$ contains a vector $(R^{i_1}, R^{i_2}, \dots, R^{i_s}, 1, 0)$ which reveals secret indexes. To hide this vector, R was chosen such that $R^{(s+1)(S-1)} > d \approx 2^{nt}$. This choice ensures that the length of this secret vector is larger than $\lambda_1(\mathcal{L}(B))$ according to Minkowski's theorem. Thus, finding the shortest vector in $\mathcal{L}(B)$ does not reveal the secret indexes. The parameters of the fully homomorphic scheme used for public challenges are shown in Table 1 [3, Table 3].

In the next section, we modify the above lattice basis to make the shortest vector reveal the secret indexes.

	Dimension n	bit-size t	sparse subset-size s	big-set size S	big-set ratio R
Toy	512	380	15	512	2^{26}
Small	2048	380	15	512	2^{102}
Medium	8192	380	15	547	2^{381}
Large	32768	380	15	2185	2^{381}

Table 1. Parameters of the fully homomorphic scheme

4 Solving the SSSP challenges

In this section, we modify the above lattice basis in two ways, one leads to a deterministic method to SSSP while the other one leads to a probabilistic method. Also, experimental results of the second method are shown since it is faster.

Recall that R was chosen such that

$$R^{S-1} > d^{1/(s+1)}. \tag{3}$$

To reverse this inequality, we increase the right-hand side in Section 4.1; while we decrease the left-hand side in Section 4.2.

4.1 Dimension reduction method

Note that i_k in equation (1) has only S possibilities. Using this fact, we use the dimension reduction method which was used to attack NTRU [6] in [9] and later against GGH [5] in [7]. Namely, for each possibility of i_1, we find the shortest vector in the lattice generated by the following basis matrix

$$B' = \begin{pmatrix} 1 & & & x_2 N \\ & \ddots & & \\ & & 1 & x_s N \\ & & & 1 \; \langle -w + x_1 R^{i_1} \rangle_d N \\ & & & dN \end{pmatrix}, \tag{4}$$

for a suitable constant N which could be set to $d^{1/s}$. At this stage, this lattice $\mathcal{L}(B')$ has the dimension $(s+1)$; for the correct index i_1, the shortest vector is expected to reveal the secret indexes if $R^{S-1} < d^{1/s}$, which is the case for the parameters in [3]. Reducing the dimension of the lattice basis by one, we could make the shortest vector of the lattice reveal the secret indexes if we know i_1. Because we do not know i_1, we should try S times.

In general, we need to choose the integer k_0 such that $R^{S-1} < d^{1/(s+1-k_0)}$ and try S^{k_0} times. In the SSSP challenges, $k_0 = 1$ is sufficient.

This clearly shows that the SSSP challenges in [3] are rather easily solvable by lattice-based attacks. In the next subsection, we present another method.

4.2 Probabilistic method

To decrase the left-hand side of equation (3), we first select the integer l_0 such that

$$R^{S-1-l_0} < d^{1/(s+1)}. \tag{5}$$

Then randomly choose non-negative integers $l_1, l_2, \ldots, l_s$ less than or equal to l_0. Assuming the secret indexes of the SSSP are generated randomly, the secret index i_k should be distributed uniformly in the set $\{0, 1, \ldots, S-1\}$. Here, we can expect that with a probability of $(S-l_0)/S$, the secret index i_k is in the interval $[l_k, l_k+S-1-l_0]$. When this situation arises, equation (1) becomes

$$\sum_k x_k \cdot R^{i_k} = \sum_k x_k R^{l_k} \cdot R^{i_k-l_k} = w \pmod d, \tag{6}$$

and the shifted secret vector $(R^{i_1-l_1}, R^{i_2-l_2}, \ldots, R^{i_s-l_s}, 1, 0)$ may be the shortest vector in the lattice generated by following basis matrix

$$B'' = \begin{pmatrix} 1 & & & & & \langle x_1 R^{l_1}\rangle_d N \\ & 1 & & & & \langle x_2 R^{l_2}\rangle_d N \\ & & \ddots & & & \\ & & & 1 & & \langle x_s R^{l_s}\rangle_d N \\ & & & & 1 & -wN \\ & & & & & dN \end{pmatrix}, \tag{7}$$

for the suitable constant N, which is set to $d^{1/(s+1)}$. With a probability of $((S-l_0)/S)^s$, the length of the shifted secret vector could be the shortest vector of $\mathcal{L}(B'')$ since

$$R^{S-1-l_0} < d^{1/(s+1)} = (dN)^{1/(s+2)}. \tag{8}$$

Thus, lattice reduction algorithms would find the secret indexes with a probability of $((S-l_0)/S)^s$.

Due to the aggressive parameter choice, a rather small l_0 is sufficient. For a small challenge, $l_0 = 35$ satisfies equation (5) and the probability of success is about 34%. For a large challenge, $l_0 = 114$ is sufficient and the success probability becomes 44%.

In the next subsection, the experimental results of this method for SSSP challenges are shown.

4.3 Experiments

For experiments, we used small and medium challenges from the homepage[1] as well as a large challenge obtained from the authors of [4]. The integer l_0 was chosen as shown in the Table 2 such that equation (5) is satisfied. And non-negative integers $l_1, \ldots, l_s$ less than l_0 are randomly chosen. Then the lattice basis B'' is reduced using the LLL algorithm in the FPLLL library [1]. In the results, we could find the secret indexes for all SSSP challenges.

Timing results for SSSP challenges are shown in the Table 2. As the table shows, even the large challenge is solved within two days. Our experiment was conducted in a CPU core i7 running at 3.40GHz. This clearly shows that the parameter choice for the SSSP in [4] was too aggressive in that it did not consider these attacks.

[1] `https://researcher.ibm.com/researcher/view_project.php?id=1548`

	sparse subset-size s	big-set size S	determinant d	l_0	lattice reduction time
Small	15	512	$\log_2 d \approx 785006$	35	7m
Medium	15	547	$\log_2 d \approx 3148249$	35	1h 49m
Large	15	2185	$\log_2 d \approx 12625500$	114	34h 22m

Table 2. Timing results for SSSP challenges

5 Discussion

To remedy our attack, it is necessary to change the parameters such that $R^{S-1} \gg d^{1/(s+1)}$. One could use larger values for s and/or R. For example, as noted in [3], using $s = 30$ is likely to increase the size of the key (and the running time) by only about 5-10%. When this choice is made, our attacks become inefficient. Our first attack, dimension reduction method needs to eliminate at least 16 rows, implying a complexity level of at least S^{16}. Our probabilistic method also requires $l_0 \approx S/2$ implying that the probability of success is only about $2^{-s} = 2^{-30}$. Thus, this seems to be a reasonable choice.

Note that our result does not imply the insecurity of the fully homomorphic encryption in [3], as we only solved a sparse subset sum problem with the knowledge of w, which is a secret key in practice. As is discussed in [12], the real attacker of homomorphic encryption scheme does not know the secret key w and would encounter what has been termed the hidden sparse subset sum problem, which we do not know how to solve at present.

6 Conclusion

In this paper, we demonstrated how to solve a specific form of the sparse subset sum problem given in [3] using lattice reduction algorithms. Our experiments show that the parameter choice in [3] is too aggressive. We also note that modifying the parameters easily remedies this situation. Although our attack is not a threat to the fully homomorphic encryption in [3] in practice, it shows that one should be careful when embedding structures into the problem, such as geometric progressions into a sparse subset sum problem in this case.

References

1. Cadé, D., Pujol, X., Stehlé, D.: FPLLL library, version 3.1.1, http://perso.ens-lyon.fr/damien.stehle
2. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.
3. Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520, 2010. `http://eprint.iacr.org/`, full version of [4].

4. Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. In Kenneth Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-20465-4_9.
5. Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 112–131, London, UK, 1997. Springer-Verlag.
6. Jeffrey Hoffstein, Jill Pipher, and Joseph Silverman. Ntru: A ring-based public key cryptosystem. In Joe Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0054868.
7. Moon Lee and Sang Hahn. Cryptanalysis of the ggh cryptosystem. *Mathematics in Computer Science*, 3:201–208, 2010. 10.1007/s11786-009-0018-5.
8. Ak Lenstra, H W Lenstra, and L Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
9. Alexander May and Joseph Silverman. Dimension reduction methods for convolution modular lattices. In Joseph Silverman, editor, *Cryptography and Lattices*, volume 2146 of *Lecture Notes in Computer Science*, pages 110–125. Springer Berlin / Heidelberg, 2001. 10.1007/3-540-44670-2_10.
10. C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66:181–199, 1994. 10.1007/BF01581144.
11. N. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Nguyen and David Pointcheval, editors, *Public Key Cryptography – PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-13013-7_25.
12. N.P. Smart and F. Vercauteren. Fully homomorphic simd operations. Cryptology ePrint Archive, Report 2011/133, 2011. `http://eprint.iacr.org/`.
13. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-13190-5_2.