Cryptanalysis of GGH Map*

Yupu Hu and Huiwen Jia

ISN Laboratory, Xidian University, 710071 Xi'an, China
yphu@mail.xidian.edu.cn hwjia@stu.xidian.edu.cn

Abstract. Multilinear map is a novel primitive which has many cryptographic applications, and GGH map is a major candidate of K-linear maps for $K > 2$. GGH map has two classes of applications, which are applications with public tools for encoding and with hidden tools for encoding. In this paper, we show that applications of GGH map with public tools for encoding are not secure, and that one application of GGH map with hidden tools for encoding is not secure. On the basis of weak-DL attack presented by the authors themselves, we present several efficient attacks on GGH map, aiming at multipartite key exchange (MKE) and the instance of witness encryption (WE) based on the hardness of exact-3-cover (X3C) problem. First, we use special modular operations, which we call modified encoding/zero-testing to drastically reduce the noise. Such reduction is enough to break MKE. Moreover, such reduction negates K-GMDDH assumption, which is a basic security assumption. The procedure involves mostly simple algebraic manipulations, and rarely needs to use any lattice-reduction tools. The key point is our special tools for modular operations. Second, under the condition of public tools for encoding, we break the instance of WE based on the hardness of X3C problem. To do so, we not only use modified encoding/zero-testing, but also introduce and solve "combined X3C problem", which is a problem that is not difficult to solve. In contrast with the assumption that multilinear map cannot be divided back, this attack includes a division operation, that is, solving an equivalent secret from a linear equation modular some principal ideal. The quotient (the equivalent secret) is not small, so that modified encoding/zero-testing is needed to reduce size. This attack is under an assumption that some two vectors are co-prime, which seems to be plausible. Third, for hidden tools for encoding, we break the instance of WE based on the hardness of X3C problem. To do so, we construct level-2 encodings of 0, which are used as alternative tools for encoding. Then, we break the scheme by applying modified encoding/zero-testing and combined X3C, where the modified encoding/zero-testing is an extended version. This attack is under two assumptions, which seem to be plausible. Finally, we present cryptanalysis of two simple revisions of GGH map, aiming at MKE. We show that MKE on these two revisions can be broken under the assumption that 2^K is polynomially large. To do so, we further extend our modified encoding/zero-testing.

Keywords: Multilinear maps, Multipartite key exchange (MKE), Witness encryption (WE), Lattice based cryptography.

1 Introduction

1.1 Background and Our Contributions

Multilinear map is a novel primitive. Mathematically speaking, multilinear map is a leveled encoding system. In other words, it is such a system that can multiply but cannot divide back, and goes further to let us recover some limited information. It is the solution of a long-standing open problem [1], and has many novel cryptographic applications, such as multipartite key exchange (MKE) [2], witness encryption (WE) [3–9], obfuscation [8–10], and so on. It also has several advantages in the traditional cryptographic area such as IBE, ABE [11], Broadcasting encryption, and so on. The first candidate of multilinear map is GGH map [2], and GGHLite map [12] is a special version of GGH map for the purpose of improving efficiency. Up until now, GGH map is a major candidate of K-linear maps for $K > 2$. It uses noisy encoding to obtain the trapdoor. The security of GGH map is not well-understood. In particular, hardness of lattice problems is necessary for its security, but it is not sufficient. GGH map has two classes of applications. The first class is applications with public tools for encoding/zero-testing such as MKE [2], IBE, ABE, Broadcasting encryption, and so on. The second class contains applications with hidden tools for encoding such as GGHRSW obfuscation [8]. WE can be in the first and second classes. For the first class, WE tools for encoding are generated and published by the system, and can be used by any user. For the second class, WE tools for encoding are generated and hidden by a unique encrypter, and can only be used by him/herself. Besides, WE is another novel cryptographic notion and the instance of WE based on the hardness of exact-3-cover (X3C) problem is its first instance. Garg et al. provided in [2] a survey of relevant cryptanalysis techniques from the literature, and also described two new attacks on GGH map. In particular they presented the weak-DL attack, which indicated that GGH map makes division possible to some extent, and which is used in our attacks as well. We emphasize, however, that they did not show how to use that attack to break any of their proposed schemes.

In this paper, we show that applications of GGH map with public tools for encoding are not secure, and that one application of GGH map with hidden tools for encoding is not secure. We present several efficient attacks on GGH map, aiming at MKE and the instance of WE based on the hardness of X3C problem. In all of our attacks we begin by using the weak-DL attack from [2] to recover an "equivalent secret" which is equal to the original secret modulo some known ideal, but is not small. Then we proceed as follows.

First, we use special modular operations, which we call modified encoding/zero-testing to drastically reduce the noise. Such reduction is enough to break MKE. Moreover, such reduction negates K-GMDDH assumption (Assumption 5.1 of

[11]), which is the security basis of the ABE scheme [11]. The procedure involves mostly simple algebraic manipulations, and rarely needs to use any lattice-reduction tools. The key point is our special tools for modular operations.

Second, under the condition of public tools for encoding, we break the instance of WE based on the hardness of X3C problem. To do so, we not only use modified encoding/zero-testing, but also introduce and solve "combined X3C problem", which is a problem that is not difficult to solve. In contrast with the assumption that multilinear map cannot be divided back, this attack includes a division operation, that is, solving an equivalent secret from a linear equation modular some principal ideal. The quotient (the equivalent secret) is not small, so that modified encoding/zero-testing is needed to reduce size. This attack is under an assumption that some two vectors are co-prime, which seems to be plausible.

Third, for hidden tools for encoding, we break the instance of WE based on the hardness of X3C problem. To do so, we construct level-2 encodings of 0, which are used as alternative tools for encoding. Then, we break the scheme by applying modified encoding/zero-testing and combined X3C, where the modified encoding/zero-testing is an extended version. This attack has several preparing works, including solving a new type of "equivalent secret". This attack is under two assumptions, which seem to be plausible.

Finally, we check whether GGH structure can be simply revised to avoid our attack. We present cryptanalysis of two simple revisions of GGH map, aiming at MKE. We show that MKE on these two revisions can be broken under the assumption that 2^K is polynomially large. To do so, we further extend our modified encoding/zero-testing.

1.2 Principles and Main Techniques of Our Attack

Quite unlike the original DH maps and bilinear maps, all candidates of multilinear maps have a common security worry that zero-testing tools are public. This allows the adversary to zero-test messages freely. The adversary can choose those zero-tested messages that are small enough without protection of the modular operation. Such security worry has been used to break CLT map [13–17], which is another major candidate of multilinear maps, and which is simply over integers. Multilinear maps over the integer polynomials (GGH map [2] and GGHLite map [12]) haven't been broken because (1) (NTRU declaration) the product of a short polynomial and modular inverse of another short polynomial seems unable to be decomposed; and (2) the product of several short polynomials seems unable to be decomposed. However, the product of several short polynomials is a somewhat short polynomial. Although it cannot be decomposed, it can be used as a modulus to reduce the noise. On the other hand, breaking applications of GGH map with public tools for encoding does not mean solving the users' secrets. It only means solving "high-order bits of zero-test of the product of encodings of users' secrets", a weaker requirement. Therefore, by using our modified encoding/zero-testing, we can easily migrate between modular operations and real number operations to find vulnerabilities which have not been

found before. All of the above form the first principle of our attack. The second principle is that if one uses GGH map for constructing the instance of WE based on the hardness of X3C problem, special structure of GGH map allows us to transform the underlying X3C problem into a much easier combined X3C problem. Our main techniques are as follows.

Modified encoding/zero-testing. For the secret of each user, we have an equivalent secret which is the sum of original secret and a noise. These equivalent secrets cannot be encoded, because they are not small. We compute the product of these equivalent secrets, rather than computing their modular product. Notice that the product is the sum of the product of original secrets and a noise. Then our modified encoding/zero-testing is quite simple. It contains three simple operations, avoiding computing original secrets of users, and extracting same information. That is, it extracts same high-order bits of zero-tested message. Table 1 is a comparison between processing routines of GGH map and our work. It is a note of our claim that we can achieve the same purpose without knowing the secret of any user.

Table 1. Processing routines

GGH map	secrets → encodings → modular product → zero-testing → high-order bits
Our work	equivalent secrets → product → modified encoding/zero-testing → high-order bits

Solving combined exact-3-cover (Combined X3C) problem. The reason that X3C problem can be transformed into a combined X3C problem is that the special structure of GGH map sometimes makes division possible. We can solve combined X3C problem with non-negligible probability and break the instance of WE based on the hardness of X3C problem for public tools of encoding.

Finding alternative encoding tools. When encoding tools are hidden, we can use redundant information to construct alternative encoding tools. For example, there are many redundant pieces beside X3C. Encodings of these redundant pieces can be composed into several level-2 encodings of 0. Only one level-2 encoding of 0 is enough to break the instance of WE based on the hardness of X3C problem for hidden tools of encoding. This technique can be adapted to other applications of GGH map, where although encoding tools are hidden, a large number of redundant information are needed to protect some secrets.

1.3 The Organization

In subsection 1.4 we review recent works related to multilinear map. In section 2 we review GGH map and two applications, MKE and the instance of WE on X3C. In section 3 we define special tools for our attack, which are special polynomials used for our modular operations. Also in this section, for the secret of each

user, we generate an equivalent secret, which is not a short vector. Immediately, we obtain an "equivalent secret" of the product of the users' secrets, which is the product of the users' equivalent secrets. In section 4 we present modified encoding/zero-testing. We show how "high-order bits of zero-test of the product of encodings of users' secrets" can be solved, so that MKE is broken. In section 5 we show how to break the instance of WE on X3C problem with public tools for encoding. In this section, we first introduce and solve "combined X3C problem", then solve "high-order bits of zero-test of the product of encodings of users' secrets". In section 6 we present an attack on the instance of WE based on the hardness of X3C problem with hidden tools for encoding. We show that this instance can be broken under several stronger assumptions. In section 7 we present cryptanalysis of two simple revisions of GGH map, aiming at MKE. We show that MKE on these two revisions can be broken under the assumption that 2^K is polynomially large. Section 8 contains other results, some considerations, and poses several questions.

1.4 Related Works

Garg et al. presented in [2] three variants, which are "asymmetric encoding", "providing zero-test security" and "avoiding principal ideals". Arita and Handa [5] presented two applications of multilinear maps: MKE with smaller communication and an instance of WE. Their WE scheme (called AH scheme) has the security claim based on the hardness of Hamilton Cycle problem. The novelty is that they used an asymmetric multilinear map over integer matrices. Bellare and Hoang [6] presented adaptive witness encryption with stronger security than soundness security, named adaptive soundness security. Garg et al. [8] presented witness encryption by using indistinguishability obfuscation and Multilinear Jigsaw Puzzle, a simplified variant of multilinear maps. Extractable witness encryption was presented [7, 9, 10]. Gentry et al. designed multilinear maps based on graph [18]. Coron et al. presented efficient attack on CLT map for hidden tools for encoding [19]. Coron et al. designed CLT15 map [20]. Then Cheon et al. [21] and Minaud and Fouque [22] broke CLT15 respectively.

2 GGH Map and Two Applications

2.1 Notations and Definitions

We denote the rational numbers by $\mathbb{Q}$ and the integers by $\mathbb{Z}$. We specify that n-dimensional vectors of $\mathbb{Q}^n$ and $\mathbb{Z}^n$ are row vectors. We consider the $2n$'th cyclotomic polynomial ring $R = \mathbb{Z}[X]/(X^n + 1)$, and identify an element $u \in R$ with the coefficient vector of the degree-$(n-1)$ integer polynomial that represents u. In this way, R is identified with the integer lattice $\mathbb{Z}^n$. We also consider the ring $R_q = R/qR = \mathbb{Z}_q[X]/(X^n + 1)$ for a (large enough) integer q. Addition in these rings is done component-wise in their coefficients, and multiplication is polynomial multiplication modulo the ring polynomial $X^n + 1$. In some cases,

we also consider the ring $\mathbb{K} = \mathbb{Q}[X]/(X^n + 1)$, which is likewise associated with the linear space $\mathbb{Q}^n$. We redefine the operation "mod q" as follows: if q is an odd, $a(\text{mod } q)$ is within $\{-(q-1)/2, -(q-3)/2, \ldots, (q-1)/2\}$; if q is an even, $a(\text{mod } q)$ is within $\{-q/2, -(q-2)/2, \ldots, (q-2)/2\}$. For $x \in R$, $\langle x \rangle = \{x \cdot u : u \in R\}$ is the principal ideal in R generated by x (alternatively, the sub-lattice of $\mathbb{Z}^n$ corresponding to this ideal). For $x \in R$, $y \in R$, $y(\text{mod } x)$ is such a vector: $y(\text{mod } x) = ax$, where each entry of a is within [-0.5, 0.5), and $y - y(\text{mod } x) \in \langle x \rangle$. We refer the readers to Babai [23].

2.2 The GGH Construction

We secretly sample a short element $g \in R$. Let $\langle g \rangle$ be the principal ideal in R. g itself is kept secret, and no "good" description of $\langle g \rangle$ is made public. Another secret element $z \in R_q$ is chosen at random, and hence is not short.

An element y is called encoding parameter, or called level-1 encoding of 1, and is set in the following description. We secretly sample a short element $a \in R$, and let $y = (1 + ag)z^{-1}(\text{mod } q)$. The elements $\{x^{(i)}, i = 1, 2\}$ are called randomizers, or called level-1 encodings of 0, and are set as follows. We secretly sample a short element $b^{(i)} \in R$, and let $x^{(i)} = b^{(i)}gz^{-1}(\text{mod } q)$, $i = 1, 2$. The public element p_{zt} is called level-K zero-testing parameter, where $K \geq 3$ is an integer. p_{zt} is set as follows. We secretly sample a "somewhat small" element $h \in R$, and let $p_{zt} = (hz^K g^{-1})(\text{mod } q)$. Simply speaking, parameters y and $\{x^{(i)}, i = 1, 2\}$ are tools for encoding, while public parameter p_{zt} is tool of zero-test. $\{g, z, a, \{b^{(i)}, i = 1, 2\}, h\}$ are kept from all users. For MKE, y and $\{x^{(i)}, i = 1, 2\}$ are public. For WE, they can be either public or hidden.

Suppose a user has a secret $v \in R$, which is a short element. He secretly samples short elements $\{u^{(i)} \in R, i = 1, 2\}$. He computes noisy encoding $V = vy + (u^{(1)}x^{(1)} + u^{(2)}x^{(2)})(\text{mod } q)$, where $vy(\text{mod } q)$ and $(u^{(1)}x^{(1)} + u^{(2)}x^{(2)})(\text{mod } q)$ are respectively encoded secret and encoded noise. He publishes V. Then, GGH K-linear map includes $K, y, \{x^{(i)}, i = 1, 2\}, p_{zt}$, and all noisy encoding Vs for all users.

We call g grade 1 element, and denote σ as the standard deviation for sampling g. We call $\{a, \{b^{(i)}, i = 1, 2\}\}$ and $\{v, \{u^{(i)}, i = 1, 2\}\}$ grade 2 elements, and denote σ' as the standard deviation for sampling $\{a, \{b^{(i)}, i = 1, 2\}\}$ and $\{v, \{u^{(i)}, i = 1, 2\}\}$. Both σ and σ' are much smaller than $\sqrt{q}$, and GGH K-linear map [2] suggests $\sigma' = n\sigma$. Finally, we call h grade 3 element, and take $\sigma'' = \sqrt{q}$ as the standard deviation for sampling h. We say that g, $\{a, \{b^{(i)}, i = 1, 2\}\}$ and $\{v, \{u^{(i)}, i = 1, 2\}\}$ are "very small", and that h is "somewhat small". h cannot be "very small" for security reasons.

2.3 Application 1: MKE

Suppose that $K + 1$ users want to generate a commonly shared key by public discussion. To do so, each user k generates his secret $v^{(k)}$, and publishes the noisy encoding $V^{(k)}, k = 1, \ldots, K + 1$. Then, each user can use his/her secret and other users' noisy encodings to compute KEY, the commonly shared

key. KEY is high-order bits of any zero-tested message. For example, user k_0 first computes $v^{(k_0)}p_{zt}\prod_{k\neq k_0}V^{(k)}(\text{mod } q)$, then KEY is high-order bits of $v^{(k_0)}p_{zt}\prod_{k\neq k_0}V^{(k)}(\text{mod } q)$. That is, he/she first computes

$$v^{(k_0)}p_{zt}\prod_{k\neq k_0}V^{(k)}(\text{mod } q) =$$
$$h(1+ag)^K g^{-1}\prod_{k=1}^{K+1}v^{(k)}+$$
$$hv^{(k_0)}\sum_{\substack{S\subset\{1,\ldots,K+1\}\\ -\{k_0\},|S|\geq 1}}(1+ag)^{K-|S|}g^{|S|-1}\prod_{\substack{k\in\{1,\ldots,K+1\}-\\ \{k_0\}-S}}(v^{(k)})\prod_{t\in S}(u^{(t,1)}b^{(1)}+u^{(t,2)}b^{(2)})(\text{mod } q).$$

It is the modular sum of two terms, zero-tested message and zero-tested noise. Zero-tested message is

$$h(1+ag)^K g^{-1}\prod_{k=1}^{K+1}v^{(k)}(\text{mod } q).$$

Zero-tested noise is

$$hv^{(k_0)}\sum_{\substack{S\subset\{1,\ldots,K+1\}\\ -\{k_0\},|S|\geq 1}}(1+ag)^{K-|S|}g^{|S|-1}\prod_{\substack{k\in\{1,\ldots,K+1\}\\ -\{k_0\}-S}}(v^{(k)})\prod_{t\in S}(u^{(t,1)}b^{(1)}+u^{(t,2)}b^{(2)}).$$

Notice that zero-tested noise is the sum of 3^K-1 terms. For example, $h(1+ag)^{K-1}b^{(1)}u^{(1,1)}\prod_{k=2}^{K+1}(v^{(k)})$ is a term of the zero-tested noise. Each term is the product of a "somewhat small" element and several "very small" elements. Therefore, zero-tested noise is "somewhat small", and it can be removed if we only extract high-order bits of $v^{(k_0)}p_{zt}\prod_{k\neq k_0}V^{(k)}(\text{mod } q)$. In other words, KEY is actually high-order bits of zero-tested message $h(1+ag)^K g^{-1}\prod_{k=1}^{K+1}v^{(k)}(\text{mod } q)$.

2.4 Application 2: The Instance of WE on Exact-3-cover

Definition 1. *A witness encryption scheme for an* **NP** *language L (with corresponding witness relation Rel) consists of the following two polynomial-time algorithms:*

Encryption. *The algorithm Encrypt(1^λ,x,M) takes as input a security parameter 1^λ, a string x, and a message M, and outputs a ciphertext* CT.

Decryption. *The algorithm Decrypt(*CT*,w) takes as input a ciphertext* CT *and a string w, and outputs a message M if Rel(w,x)=1 or the symbol $\perp$ otherwise.*

exact-3-cover problem [3, 24]. If we are given a subset of $\{1,2,\ldots,3K\}$ containing 3 integers, we call it a piece. If we are given a collection of K pieces

without intersection, we call it a X3C of $\{1, 2, \ldots, 3K\}$. The X3C problem is that for arbitrarily given $N(K)$ different pieces with a hidden X3C, find it. It is clear that $1 \leq N(K) \leq C_{3K}^3$. Intuitively, the X3C problem is often not hard when $N(K) \leq O(K)$, because X3C is not hidden well. An extreme example is that if the number i is contained by only one piece $\{i, j, k\}$, then $\{i, j, k\}$ is certainly from X3C. Picking up $\{i, j, k\}$ and abandoning those pieces containing j or k, then other pieces form a reduced X3C problem on $\{1, 2, \ldots, 3K\} - \{i, j, k\}$. So that $N(K) \geq O(K^2)$ to avoid weak case. On the other hand, the larger $N(K)$ the easier our attack. So that in rest of this paper we will always take $N(K) = O(K^2)$.

Now we describe the WE based on the hardness of X3C problem from GGH structure.

Encryption. The encrypter samples short elements $v^{(1)}, v^{(2)}, \ldots, v^{(3K)} \in R$. He/she computes the encryption key as follows. He/she first computes $v^{(1)}v^{(2)} \ldots v^{(3K)}y^K p_{zt} (\mathrm{mod}\ q)$, then takes $EKEY$ as its high-order bits. In fact, $EKEY$ is high-order bits of $v^{(1)}v^{(2)} \ldots v^{(3K)}(1+ag)^K hg^{-1} (\mathrm{mod}\ q)$. He/she can use $EKEY$ and an encryption algorithm to encrypt any plaintext. Then, he/she hides $EKEY$ into pieces as follows. He/she arbitrarily generates $N(K)$ different pieces of $\{1, 2, \ldots, 3K\}$, with a hidden X3C called XC. For each piece $\{i_1, i_2, i_3\}$, he/she computes noisy encoding of the product $v^{(i_1)}v^{(i_2)}v^{(i_3)}$, that is, secretly samples short elements $\{u^{(\{i_1,i_2,i_3\},i)} \in R, i = 1, 2\}$, then computes and publishes $V^{\{i_1,i_2,i_3\}} = v^{(i_1)}v^{(i_2)}v^{(i_3)}y + (u^{(\{i_1,i_2,i_3\},1)}x^{(1)} + u^{(\{i_1,i_2,i_3\},2)}x^{(2)}) (\mathrm{mod}\ q)$.

Decryption. The one who knows XC computes the zero-test of $\prod_{\{i_1,i_2,i_3\} \in XC} V^{\{i_1,i_2,i_3\}} (\mathrm{mod}\ q)$, that is, he/she computes $p_{zt} \prod_{\{i_1,i_2,i_3\} \in XC} V^{\{i_1,i_2,i_3\}} (\mathrm{mod}\ q)$. Then, $EKEY$ is its high-order bits. In other words, $p_{zt} \prod_{\{i_1,i_2,i_3\} \in XC} V^{\{i_1,i_2,i_3\}} (\mathrm{mod}\ q)$ is the modular sum of two terms, the first term is zero-tested message $v^{(1)}v^{(2)} \ldots v^{(3K)}(1+ag)^K hg^{-1} \ (\mathrm{mod}\ q)$, while the second term is zero-tested noise which doesn't affect high-order bits of $p_{zt} \prod_{\{i_1,i_2,i_3\} \in XC} V^{\{i_1,i_2,i_3\}} (\mathrm{mod}\ q)$.

3 Weak-DL Attack: Generating Equivalent Secrets

As the start of our attack, we will find equivalent secrets. The method is weak-DL attack [2].

3.1 Generating an Equivalent Secret for One User

We can obtain special elements $\{Y, X^{(i)}, i = 1, 2\}$, where

$$\begin{aligned} &Y = y^{K-1}x^{(1)}p_{zt} (\mathrm{mod}\ q) = h(1+ag)^{K-1}b^{(1)}, \\ &X^{(i)} = y^{K-2}x^{(i)}x^{(1)}p_{zt} (\mathrm{mod}\ q) = h(1+ag)^{K-2}(b^{(i)}g)b^{(1)}, \\ &i = 1, 2. \end{aligned}$$

Notice that the right sides of these equations have no operation "mod q". More precisely, each of $\{Y, X^{(i)}, i = 1, 2\}$ is a factor of a term of zero-tested noise.

For example, $Yu^{(1,1)}\prod_{k=2}^{K+1}(v^{(k)})$ is a term of the zero-tested noise. Therefore, each of $\{Y, X^{(i)}, i = 1, 2\}$ is far smaller than a term of the zero-tested noise. However, they are not small enough because of the existence of the factor h. We say they are "somewhat small", and take them as our tools.

Take the noisy encoding V (corresponding to the secret v and unknown $\{u^{(1)}, u^{(2)}\}$), and compute special element

$$W = Vy^{K-2}x^{(1)}p_{zt} (\text{mod } q) = vY + (u^{(1)}X^{(1)} + u^{(2)}X^{(2)}).$$

Notice that the right side of this equation has no operation "mod q". Then, compute

$$W(\text{mod } Y) = \big(u^{(1)}X^{(1)}(\text{mod } Y) + u^{(2)}X^{(2)}(\text{mod } Y)\big)(\text{mod } Y).$$

Step 1. By knowing $W(\text{mod } Y)$ and $\{X^{(1)}(\text{mod } Y), X^{(2)}(\text{mod } Y)\}$, we obtain $W' \in \langle X^{(i)}, i = 1, 2\rangle$ such that $W - W'(\text{mod } Y) = 0$. This is quite easy algebra, and we present the details in Appendix A. Notice that $W - W'$ is not a short vector. Denote $W' = u'^{(1)}X^{(1)} + u'^{(2)}X^{(2)}$.

Step 2. Compute $v^{(0)} = (W - W')/Y$ (division over real numbers with the quotient which is an integer vector). Then,

$$\begin{aligned} v^{(0)} &= v + ((u^{(1)}X^{(1)} + u^{(2)}X^{(2)}) - W')/Y \\ &= v + ((u^{(1)} - u'^{(1)})X^{(1)} + (u^{(2)} - u'^{(2)})X^{(2)})/Y \\ &= v + ((u^{(1)} - u'^{(1)})b^{(1)} + (u^{(2)} - u'^{(2)})b^{(2)})g/(1 + ag). \end{aligned}$$

By considering another fact that g and $1+ag$ are co-prime, we have $v^{(0)} - v \in \langle g\rangle$. We call $v^{(0)}$ an equivalent secret of v, and call residual vector $v^{(0)} - v$ the noise. Notice that $v^{(0)}$ is not a short vector.

3.2 Generating an Equivalent Secret for the Product of Secrets

Suppose that each user k has his/her secret $v^{(k)}$ and we generate $v^{(0,k)}$, an equivalent secret of $v^{(k)}$, where $k = 1, \ldots, K+1$. For the product $\prod_{k=1}^{K+1} v^{(k)}$, we have an equivalent secret $\prod_{k=1}^{K+1} v^{(0,k)}$, where the noise is $\prod_{k=1}^{K+1} v^{(0,k)} - \prod_{k=1}^{K+1} v^{(k)} \in \langle g\rangle$. Notice that $\prod_{k=1}^{K+1} v^{(0,k)}$ is not a short vector.

4 Modified Encoding/Zero-testing

In this section we transform $\prod_{k=1}^{K+1} v^{(0,k)}$ by our modified encoding/zero-testing. Denote $\eta = \prod_{k=1}^{K+1} v^{(0,k)}$. The procedure has three steps, which are $\eta' = Y\eta$, $\eta'' = \eta'(\text{mod } X^{(1)})$, and $\eta''' = y(x^{(1)})^{-1}\eta''(\text{mod } q)$ (or $\eta''' = Y(X^{(1)})^{-1}\eta''(\text{mod } q)$). To help understanding their functions, we compare them with GGH processing procedure. The first operation is like a level-K encoding followed by a zero-testing, but there are three differences. Difference 1: The first operation doesn't use modular q. Difference 2: $\eta'(\text{mod } q)$ contains a modular q factor y^{K-1}, while

zero-tested message contains a modular q factor y^K. In other words, $\eta'(\text{mod } q)$ lacks a y. Difference 3: $\eta'(\text{mod } q)$ contains a modular q factor $x^{(1)}$, while zero-tested message doesn't contain such modular q factor. In other words, $\eta'(\text{mod } q)$ has a surplus $x^{(1)}$. η'' is also like a level-K encoding followed by a zero-testing, and there are also three differences as above, but the size is reduced to "somewhat small". To obtain η''', we get rid of $x^{(1)}$ and put y in so that η''' is a level-K encoding followed by a zero-testing, and that we can guarantee zero-tested noise "somewhat small". Notice $\eta = \prod_{k=1}^{K+1} v^{(k)} + \xi g$, where $\xi \in R$.

Step 1. Compute $\eta' = Y\eta$. By noticing that Y is a multiple of $b^{(1)}$, we have a fact that $\eta' = Y\prod_{k=1}^{K+1} v^{(k)} + \xi' b^{(1)} g$, where $\xi' \in R$.

Step 2. Compute $\eta'' = \eta'(\text{mod } X^{(1)})$. There are 3 facts as follows.

(1) $\eta'' = Y\prod_{k=1}^{K+1} v^{(k)} + \xi'' b^{(1)} g$, where $\xi'' \in R$. Notice that η'' is the sum of η' and a multiple of $X^{(1)}$, and that $X^{(1)}$ is a multiple of $b^{(1)}g$.

(2) η'' has a similar size to that of $\sqrt{n}X^{(1)}$. In other words, η'' is smaller than one term of zero-tested noise. Notice standard deviations for sampling various variables.

(3) $Y\prod_{k=1}^{K+1} v^{(k)}$ has a similar size to that of one term of zero-tested noise.

The above 3 facts result in a new fact that $\xi'' b^{(1)} g = \eta'' - Y\prod_{k=1}^{K+1} v^{(k)}$ has a similar size to that of one term of zero-tested noise.

Step 3. Compute $\eta''' = y(x^{(1)})^{-1}\eta''(\text{mod } q)$. There are 3 facts as follows.

(1) $\eta''' = (h(1+ag)^K g^{-1})\prod_{k=1}^{K+1} v^{(k)} + \xi''(1+ag)(\text{mod } q)$. Notice fact (1) of Step 2, and notice the definitions of Y and $X^{(1)}$.

(2) $\xi''(1+ag)$ has a similar size to that of one term of zero-tested noise. In other words, $\xi''(1+ag)$ is smaller than zero-tested noise. This fact is clear by noticing that $\xi'' b^{(1)} g$ has a similar size to that of one term of zero-tested noise, and by noticing that $1+ag$ and $b^{(1)}g$ have a similar size.

(3) $(h(1+ag)^K g^{-1})\prod_{k=1}^{K+1} v^{(k)}(\text{mod } q)$ is zero-tested message, therefore its high-order bits are what we want to obtain.

The above 3 facts result in a new fact that η''' is the modular sum of zero-tested message and a new zero-tested noise which is smaller than original zero-tested noise. Therefore, high-order bits of η''' are what we want to obtain. MKE has been broken. More important is that K-GMDDH assumption (Assumption 5.1 of [11]) is negated.

5 Breaking the Instance of WE Based on the Hardness of Exact-3-cover Problem with Public Tools for Encoding

Our modified encoding/zero-testing cannot directly break the instance of WE based on the hardness of X3C problem, because the X3C is hidden. In this section we show that special structure of GGH map can simplify the X3C problem into a combined X3C problem, and then show how to use a combined exact cover to break the instance under the condition that low-level encodings of zero are made publicly available.

5.1 Combined Exact-3-cover Problem: Definition and Solution

Definition 2. *Suppose we are given $N(K) = O(K^2)$ different pieces of $\{1, 2, \ldots, 3K\}$. A subset $\{i_1, i_2, i_3\}$ of $\{1, 2, \ldots, 3K\}$ is called a combined piece, if*

(1) $\{i_1, i_2, i_3\}$ is not a piece;
(2) $\{i_1, i_2, i_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{l_1, l_2, l_3\}$;
(3) $\{j_1, j_2, j_3\}$, $\{k_1, k_2, k_3\}$ and $\{l_1, l_2, l_3\}$ are pieces;
(4) $\{j_1, j_2, j_3\}$ and $\{k_1, k_2, k_3\}$ don't intersect. (Then $\{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} \supset \{l_1, l_2, l_3\}$).

Definition 3. *A subset $\{i_1, i_2, i_3\}$ of $\{1, 2, \ldots, 3K\}$ is called a second-order combined piece, if*

(1) $\{i_1, i_2, i_3\}$ is neither a piece nor a combined piece;
(2) $\{i_1, i_2, i_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{l_1, l_2, l_3\}$;
(3) $\{j_1, j_2, j_3\}$, $\{k_1, k_2, k_3\}$ and $\{l_1, l_2, l_3\}$ are pieces or combined pieces.
(4) $\{j_1, j_2, j_3\}$ and $\{k_1, k_2, k_3\}$ don't intersect. (Then $\{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} \supset \{l_1, l_2, l_3\}$).

K pieces or combined pieces or second-order combined pieces without intersection are called a combined X3C of $\{1, 2, \ldots, 3K\}$. The combined X3C problem is that for arbitrarily given $N(K) = O(K^2)$ different pieces, find a combined X3C. We will show that the combined X3C problem is not difficult to solve. More specifically, suppose that $O(K^2)$ pieces are sufficiently randomly distributed, in them there is a hidden X3C, and the instance of X3C problem is assumed to be hard. Then we will prove that corresponding instance of combined X3C problem can be solved in polynomial time. Our proving procedure has two steps, which are obtaining combined pieces and obtaining second-order combined pieces.

Obtaining combined pieces. We take $P(E)$ as the probability of the event E, and $P(E|E')$ as the conditional probability of E under the condition E'. Arbitrarily take a subset $\{i_1, i_2, i_3\}$ which is not a piece. In Appendix B we show that $P(\{i_1, i_2, i_3\}$ is not a combined piece$) \approx exp\{-(O(K^2))^3/K^6\}$. For the sake of simple deduction, we temporarily assume $O(K^2) > K^2$, then this probability is smaller than e^{-1}. Now we construct all combined pieces from $O(K^2)$ pieces, and we have a result: there are more than $(1 - e^{-1})C_{3K}^3$ different subsets of $\{1, 2, \ldots, 3K\}$, each containing 3 elements, which are pieces or combined pieces.

Obtaining second-order combined pieces. There are less than $e^{-1}C_{3K}^3$ different subsets of $\{1, 2, \ldots, 3K\}$, each containing 3 elements, which are neither pieces nor combined pieces. Arbitrarily take one subset $\{i_1, i_2, i_3\}$ from them. By a deduction procedure similar to Appendix B, we can show that $P(\{i_1, i_2, i_3\}$ is not a second-order combined piece$)$ is negatively exponential in K. Now we construct all second-order combined pieces from more than $(1 - e^{-1})C_{3K}^3$ pieces or combined pieces, and then we are almost sure to have a result: all C_{3K}^3 different subsets of $\{1, 2, \ldots, 3K\}$, each containing 3 elements, are pieces or combined pieces or second-order combined pieces. Therefore, the combined X3C problem is solved.

5.2 Positive/Negative Factors

Definition 4. *Take a fixed combined X3C. Take an element* $\{i_1, i_2, i_3\}$ *of this combined X3C.*

(1) If $\{i_1, i_2, i_3\}$ *is a piece, we count it as a positive factor.*

(2) If $\{i_1, i_2, i_3\}$ *is a combined piece,* $\{i_1, i_2, i_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{l_1, l_2, l_3\}$*, we count pieces* $\{j_1, j_2, j_3\}$ *and* $\{k_1, k_2, k_3\}$ *as positive factors, and count the piece* $\{l_1, l_2, l_3\}$ *as a negative factor.*

(3) Suppose $\{i_1, i_2, i_3\}$ *is a second-order combined piece,*$\{i_1, i_2, i_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{l_1, l_2, l_3\}$*, where* $\{j_1, j_2, j_3\}$*,* $\{k_1, k_2, k_3\}$ *and* $\{l_1, l_2, l_3\}$ *are pieces or combined pieces.*

(3.1) If $\{j_1, j_2, j_3\}$ *is a piece, we count it as a positive factor; if* $\{j_1, j_2, j_3\}$ *is a combined piece, we count 2 positive factors corresponding to it as positive factors, and the negative factor corresponding to it as a negative factor.*

(3.2) Similarly, if $\{k_1, k_2, k_3\}$ *is a piece, we count it as a positive factor; if* $\{k_1, k_2, k_3\}$ *is a combined piece, we count 2 positive factors corresponding to it as positive factors, and the negative factor corresponding to it as a negative factor.*

(3.3) Oppositely, if $\{l_1, l_2, l_3\}$ *is a piece, we count it as a negative factor; if* $\{l_1, l_2, l_3\}$ *is a combined piece, we count 2 positive factors corresponding to it as negative factors, and the negative factor corresponding to it as a positive factor.*

Positive and negative factors are pieces. All positive factors form a collection, and all negative factors form another collection (notice that we use the terminology "collection" rather than "set", because it is possible that one piece is counted several times). Take CPF as the collection of positive factors, NPF as the number of positive factors. Take CNF as the collection of negative factors, NNF as the number of negative factors. Notice that some pieces may be counted repeatedly. It is easy to see that $NPF - NNF = K$. On the other hand, from C_{3K}^3 different subsets of $\{1, 2, \ldots, 3K\}$, there are $O(K^2)$ different pieces, more than $(1 - e^{-1})C_{3K}^3 - O(K^2)$ different combined pieces, and less than $e^{-1}C_{3K}^3$ different second-order combined pieces. Each piece is a positive factor, each combined piece is attached by 2 positive factors and a negative factor, each second-order combined piece is attached by at most 5 positive factors and 4 negative factors. Therefore, for a randomly chosen combined X3C, it is almost sure that $NPF \leq 3K$, resulting in $NNF \leq 2K$.

5.3 Our Construction

Randomly take a combined X3C. Obtain CPF, the collection of positive factors, and CNF, the collection of negative factors. For a positive factor $pf = \{i_1, i_2, i_3\}$, we denote $v^{(pf)} = v^{(i_1)}v^{(i_2)}v^{(i_3)}$ as the secret of pf, and $v'^{(pf)}$ as the equivalent secret of $v^{(pf)}$ obtained in subsection 3.1. Similarly we denote $v^{(nf)}$ and

$v'^{(nf)}$ for a negative factor nf. Denote $PPF = \prod_{pf \in CPF} v'^{(pf)}$ as the product of equivalent secrets of all positive factors. Denote $PNF = \prod_{nf \in CNF} v'^{(nf)}$ as the product of equivalent secrets of all negative factors. Denote $PTS = \prod_{k=1}^{3K} v^{(k)}$ as the product of true secrets. The first clear equation is $\prod_{pf \in CPF} v^{(pf)} = PTS \times \prod_{nf \in CNF} v^{(nf)}$. Then, we have

Proposition 1.

(1) $PPF - \prod_{pf \in CPF} v^{(pf)} \in \langle g \rangle$.
(2) $PNF - \prod_{nf \in CNF} v^{(nf)} \in \langle g \rangle$.
(3) $PPF - PNF \times PTS \in \langle g \rangle$.

Proof. By considering subsection 3.1, we know that

(1) $PPF = \prod_{pf \in CPF} v^{(pf)} + \beta_{PF}$, where $\beta_{PF} \in \langle g \rangle$.
(2) $PNF = \prod_{nf \in CNF} v^{(nf)} + \beta_{NF}$, where $\beta_{NF} \in \langle g \rangle$.

On the other hand, (3) is true from

$$\prod_{pf \in CPF} v^{(pf)} = PTS \times \prod_{nf \in CNF} v^{(nf)}.$$

Proposition 1 is proven. □

Perhaps there is hope in solving PTS. However, we cannot filter off β_{PF} and β_{NF}, because no "good" description of $\langle g \rangle$ has been made public. Fortunately, we don't need to solve PTS for breaking the instance. We only need to find an equivalent secret of PTS, without caring about the size of the equivalent secret. Then, we can reduce zero-tested noise much smaller by our modified encoding/zero-testing. Proposition 2 describes the shape of the equivalent secret of PTS under an assumption.

Proposition 2.

(1) If PTS' is an equivalent secret of PTS, then $PPF - PNF \times PTS' \in \langle g \rangle$.
(2) Assume that PNF and g are co-prime. If $PPF - PNF \times PTS' \in \langle g \rangle$, then PTS' is an equivalent secret of PTS.

Proof. (1) is clear by considering (3) of Proposition 1. If $PPF - PNF \times PTS' \in \langle g \rangle$, then $PNF \times (PTS' - PTS) \in \langle g \rangle$. According to our assumption, we have $(PTS' - PTS) \in \langle g \rangle$, hence (2) is proven. □

Now we want to find an equivalent secret of PTS. From viewpoint of multilinear map, this is a division operation: We "divide" PPF by PNF to obtain PTS'. Under our assumption, we only need to find a vector $PTS' \in R$ such that $PPF - PNF \times PTS' \in \langle g \rangle$ without caring about the size of PTS'. To do so we only need to obtain a "bad" description of $\langle g \rangle$. That is, we only need to obtain a public basis of the lattice $\langle g \rangle$; for example, the Hermite normal form. This is not a difficult task, and in Appendix C we will present our method for doing so.

After obtaining a public basis G, the condition $PPF - PNF \times PTS' \in \langle g \rangle$ is transformed into an equivalent condition

$$PPF \times G^{-1} - PTS' \times \overline{PNF} \times G^{-1} \in R,$$

where G^{-1} is the inverse matrix of G, and

$$\overline{PNF} = \begin{bmatrix} PNF_0 & PNF_1 & \cdots & PNF_{n-1} \\ -PNF_{n-1} & PNF_0 & \cdots & PNF_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ -PNF_1 & -PNF_2 & \cdots & PNF_0 \end{bmatrix}.$$

Take each entry of $PPF \times G^{-1}$ and $\overline{PNF} \times G^{-1}$ as the form of reduced fraction, and take lcm as the least common multiple of all denominators, and then the condition is transformed into another equivalent condition

$$\begin{aligned} &(lcm \times PPF \times G^{-1}) (\text{mod } lcm) \\ &= PTS' \times (lcm \times \overline{PNF} \times G^{-1}) (\text{mod } lcm). \end{aligned}$$

This is a linear equation modular lcm, and it is easy to obtain a solution PTS'. After that we take our modified encoding/zero-testing, exactly the same as in section 4. Denote $\eta = PTS'$. Compute $\eta' = Y\eta$. Compute $\eta'' = \eta' (\text{mod } X^{(1)})$. Compute $\eta''' = y(x^{(1)})^{-1}\eta'' (\text{mod } q)$. Then, high-order bits of η''' are what we want to obtain. The instance has been broken.

We can explain that temporary assumption $O(K^2) > K^2$ is not needed for a successful attack. For smaller number of pieces, we can always generate combined pieces, second-order combined pieces, third-order combined pieces, ..., step by step, until we can easily obtain a combined X3C. From this combined X3C, each set is a piece or a combined piece or a second-order combined piece or a third-order combined piece or ..., rather than only a piece or a combined piece or a second-order combined piece. Then, we can obtain all positive and negative factors, which can be defined step by step. In other words, we can sequentially define positive/negative factors attached to a third-order combined piece, to a fourth-order combined piece, ..., and so on. Finally, we can break the instance by using the same procedure. The difference is merely a more complicated description. A question left is whether the assumption "PNF and g are co-prime" is a plausible case. It means that g and each factor of PNF are co-prime. The answer is seemingly yes. A test which we haven't run is that we take two different combined X3Cs, so that we obtain two different values of PNF. If they finally obtain the same high-order bits of η''', we can believe the assumption is true for two values of PNF.

6 Breaking the Instance of WE Based on the Hardness of Exact-3-cover Problem with Hidden Tools for Encoding

6.1 Preparing Work (1): Finding Level-2 Encodings of 0

Take two pieces $\{i_1, i_2, i_3\}$ and $\{j_1, j_2, j_3\}$ which do not intersect. From other pieces, randomly choose two pieces $\{k_1, k_2, k_3\}$ and $\{l_1, l_2, l_3\}$, then the prob-

ability that $\{k_1, k_2, k_3\} \cup \{l_1, l_2, l_3\} = \{i_1, i_2, i_3\} \cup \{j_1, j_2, j_3\}$ is about $\frac{1}{C_{3K}^6}$, which is polynomially small. From all of $N(K) = O(K^2)$ pieces, we construct all sets of 4 pieces, and we estimate the average number of such sets of 4 pieces $\{\{i_1, i_2, i_3\}, \{j_1, j_2, j_3\}, \{k_1, k_2, k_3\}, \{l_1, l_2, l_3\}\}$ that $\{i_1, i_2, i_3\}$ and $\{j_1, j_2, j_3\}$ do not intersect, and $\{k_1, k_2, k_3\} \cup \{l_1, l_2, l_3\} = \{i_1, i_2, i_3\} \cup \{j_1, j_2, j_3\}$. This number is of the order of magnitude $\frac{C_{O(K^2)}^4}{C_{3K}^6}$, meaning that we have "many" such sets. At least finding one such set is noticeable. Take one of such sets $\{\{i_1, i_2, i_3\}, \{j_1, j_2, j_3\}, \{k_1, k_2, k_3\}, \{l_1, l_2, l_3\}\}$ and corresponding encodings $\{V^{\{i_1,i_2,i_3\}}, V^{\{j_1,j_2,j_3\}}, V^{\{k_1,k_2,k_3\}}, V^{\{l_1,l_2,l_3\}}\}$, then

$$\left(V^{\{i_1,i_2,i_3\}} V^{\{j_1,j_2,j_3\}} - V^{\{k_1,k_2,k_3\}} V^{\{l_1,l_2,l_3\}}\right) (\text{mod } q) = ugz^{-2} (\text{mod } q),$$

where u is very small. We call it a level-2 encoding of 0. According to the statement above, we have "many" level-2 encodings of 0. Here we fix and remember one such encoding of 0, and call it V^*. Correspondingly, we fix and remember u.

6.2 Preparing Work (2): Supplement and Division

Take a combined X3C. Obtain CPF and CNF, collections of positive and negative factors. Suppose $NPF \leq 2K-2$ (therefore $NNF = NPF-K \leq K-2$. It is easy to see that this case is noticeable). Take a piece $\{i_1, i_2, i_3\}$ and supplement it $2K - NPF$ times into CPF, so that we have new $NPF = 2K$. Similarly, supplement such a piece $\{i_1, i_2, i_3\}$ $K - NNF = 2K - NPF$ times into CNF, so that we have a new $NNF = K$. We fix and remember the piece $\{i_1, i_2, i_3\}$.

Then, we divide the collection CPF into two subcollections, $CPF(1)$ and $CPF(2)$, where

(1) $\|CPF(1)\| = \|CPF(2)\| = K$. That is, $CPF(1)$ and $CPF(2)$ are of equal size.
(2) $CPF(2)$ contains $\{i_1, i_2, i_3\}$ at least twice.
(3) $CPF(1)$ contains two pieces $\{j_1, j_2, j_3\}$ and $\{k_1, k_2, k_3\}$ which do not intersect. We fix and remember these two pieces $\{j_1, j_2, j_3\}$ and $\{k_1, k_2, k_3\}$.

The purpose of such supplementation and division is the convenience for level-K zero-testing.

6.3 Preparing Work (3): Constructing the Equation

We have fixed and remembered five elements: V^* (a level-2 encoding of 0),u ($V^* = ugz^{-2} (\text{mod } q)$), $\{i_1, i_2, i_3\}$ (a piece contained by $CPF(2)$ at least twice), $\{j_1, j_2, j_3\}$ and $\{k_1, k_2, k_3\}$ (they are from $CPF(1)$, and do not intersect each other). Now we denote four elements as follows.

$$Dec(P(1)) = p_{zt} V^* \prod_{pf \in CPF(1) - \{\{j_1,j_2,j_3\},\{k_1,k_2,k_3\}\}} V^{(pf)} (\text{mod } q),$$

$$Dec(P(2)) = p_{zt}V^* \prod_{pf\in CPF(2)-\{\{i_1,i_2,i_3\},\{i_1,i_2,i_3\}\}} V^{(pf)} (\text{mod } q),$$
$$Dec(N) = p_{zt}V^* \prod_{nf\in CNF-\{\{i_1,i_2,i_3\},\{i_1,i_2,i_3\}\}} V^{(nf)} (\text{mod } q),$$
$$Dec(Original) = hV^*g^{-1}z^2 \prod_{k\in\{1,\ldots,3K\}-\{j_1,j_2,j_3,k_1,k_2,k_3\}} v^{(k)} (\text{mod } q).$$

We can rewrite $Dec(P(1))$, $Dec(P(2))$, $Dec(N)$, $Dec(Original)$, as follows.

$$Dec(P(1)) = hu \prod_{pf\in CPF(1)-\{\{j_1,j_2,j_3\},\{k_1,k_2,k_3\}\}} (v^{(pf)}(1+ag) + u^{(pf,1)}b^{(1)}g + u^{(pf,2)}b^{(2)}g),$$
$$Dec(P(2)) = hu \prod_{pf\in CPF(2)-\{\{i_1,i_2,i_3\},\{i_1,i_2,i_3\}\}} (v^{(pf)}(1+ag) + u^{(pf,1)}b^{(1)}g + u^{(pf,2)}b^{(2)}g),$$
$$Dec(N) = hu \prod_{nf\in CNF-\{\{i_1,i_2,i_3\},\{i_1,i_2,i_3\}\}} (v^{(nf)}(1+ag) + u^{(nf,1)}b^{(1)}g + u^{(nf,2)}b^{(2)}g),$$
$$Dec(Original) = hu \prod_{k\in\{1,\ldots,3K\}-\{j_1,j_2,j_3,k_1,k_2,k_3\}} v^{(k)}.$$

Notice that $\{a, b^{(1)}, b^{(2)}\}$ has been fixed and remembered in subsection 2.2. Four facts about $\{Dec(P(1)), Dec(P(2)), Dec(N), Dec(Original)\}$ are as follows.

(1) They are all somewhat small.

(2) $Dec(P(1))$, $Dec(P(2))$, $Dec(N)$ can be obtained, while $Dec(Original)$ cannot.

(3) We have the equation

$$Dec(P(1)) \times Dec(P(2)) - Dec(N) \times Dec(Original) \in \langle (hu)^2 g\rangle \subset \langle hu^2 g\rangle.$$

This equation is clear by considering the encoding procedure and definitions of $\{Dec(P(1)), Dec(P(2)), Dec(N), Dec(Original)\}$.

(4) Conversely, suppose there is $D' \in R$ such that

$$Dec(P(1)) \times Dec(P(2)) - Dec(N) \times D' \in \langle hu^2 g\rangle.$$

Then, D' is the sum of $Dec(Original)$ and an element of $\langle ug\rangle$. Here we use a small assumption that $\frac{Dec(N)}{u}$ and (ug) are co-prime, which is noticeable. In other words, D' is a solution of the equation

$$Dec(P(1)) \times Dec(P(2)) \equiv Dec(N) \times D' (\text{mod } \langle hu^2 g\rangle),$$

if and only if D' is the sum of $Dec(Original)$ and an element of $\langle ug\rangle$. Here "mod $\langle hu^2 g\rangle$" is general lattice modular operation by using a basis of the lattice $\langle hu^2 g\rangle$. We call D' "an equivalent secret" of $Dec(Original)$. Notice that such new type of "equivalent secret" and original secret are congruent modular $\langle ug\rangle$ rather than modular $\langle g\rangle$.

6.4 Solving the Equation: Finding "An Equivalent Secret"

We want to obtain "an equivalent secret" of $Dec(Original)$ without caring about the size. To do so we only need to obtain a basis of the lattice $\langle hu^2g\rangle$ (the "bad" basis). If we can obtain many elements of $\langle hu^2g\rangle$ which are somewhat small, obtaining a basis of $\langle hu^2g\rangle$ is not hard work. Arbitrarily take $K-4$ pieces $\{piece(1), piece(2), \ldots, piece(K-4)\}$ without caring whether they are repeated. Then,

$$p_{zt}(V^*)^2 \prod_{k=1}^{K-4} V^{(piece(k))} (\text{mod } q) =$$

$$hu^2g \prod_{k=1}^{K-4} (v^{(piece(k))}(1+ag) + u^{(piece(k),1)}b^{(1)}g + u^{(piece(k),2)}b^{(2)}g) \in \langle hu^2g\rangle.$$

Thus, we can generate enough elements of $\langle hu^2g\rangle$ which are somewhat small. This fact implies that finding a D' may be easy.

6.5 Reducing the Zero-tested Noise Much Smaller

Suppose we have obtained D', "an equivalent secret" of $Dec(Original)$. D' is the sum of $Dec(Original)$ and an element of $\langle ug\rangle$, and D' is not a short vector. Arbitrarily take an element of $\langle hu^2g\rangle$ which is somewhat small, and call it V^{**}. Compute $V^{***} = D' (\text{mod } V^{**})$. Two facts about V^{***} are as follows.

(1) $V^{***} = Dec(Original) + V^{****}$, where $V^{****} \in \langle ug\rangle$.

(2) Both V^{***} and $Dec(Original)$ are somewhat small, so that V^{****} is somewhat small.

Then, compute

$$V^{\#} = V^{***}V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}(V^*)^{-1} (\text{mod } q) =$$

$$\Big[\Big(Dec(Original) \times V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}(V^*)^{-1}\Big) + \Big(V^{****} \times V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}(V^*)^{-1}\Big)\Big] (\text{mod } q).$$

Two facts about $V^{\#}$ are as follows.

(1)

$$\begin{aligned}
&\Big(Dec\,(Original) \times V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}(V^*)^{-1}\Big) (\text{mod } q) \\
&= hg^{-1}V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}z^2 \prod_{k\in\{1,\ldots,3K\}-\{j_1,j_2,j_3,k_1,k_2,k_3\}} v^{(k)} (\text{mod } q) \\
&= hg^{-1}(v^{(j_1,j_2,j_3)}(1+ag) + u^{((j_1,j_2,j_3),1)}b^{(1)}g + u^{((j_1,j_2,j_3),2)}b^{(2)}g) \\
&\quad (v^{(k_1,k_2,k_3)}(1+ag) + u^{((k_1,k_2,k_3),1)}b^{(1)}g + u^{((k_1,k_2,k_3),2)}b^{(2)}g) \\
&\quad \prod_{k\in\{1,\ldots,3K\}-\{j_1,j_2,j_3,k_1,k_2,k_3\}} v^{(k)} \qquad (\text{mod } q)
\end{aligned}$$

Therefore, its high-order bits are the secret key.

(2)

$$\begin{aligned}&\left(V^{****}\times V^{(j_1,j_2,j_3)}V^{(k_1,k_2,k_3)}(V^*)^{-1}\right)(\text{mod } q)\\ =\ & V^{****}(ug)^{-1}(v^{(j_1,j_2,j_3)}(1+ag)+u^{((j_1,j_2,j_3),1)}b^{(1)}g+u^{((j_1,j_2,j_3),2)}b^{(2)}g)\\ & (v^{(k_1,k_2,k_3)}(1+ag)+u^{((k_1,k_2,k_3),1)}b^{(1)}g+u^{((k_1,k_2,k_3),2)}b^{(2)}g) \quad (\text{mod } q).\end{aligned}$$

It is somewhat small because V^{****} is somewhat small, V^{****} is a multiple of (ug), and (ug) and

$$\begin{aligned}&(v^{(j_1,j_2,j_3)}(1+ag)+u^{((j_1,j_2,j_3),1)}b^{(1)}g+u^{((j_1,j_2,j_3),2)}b^{(2)}g)\times\\ &(v^{(k_1,k_2,k_3)}(1+ag)+u^{((k_1,k_2,k_3),1)}b^{(1)}g+u^{((k_1,k_2,k_3),2)}b^{(2)}g)\end{aligned}$$

have same size.

These two facts mean that high-order bits of $V^{\#}$ are the secret key. The instance has been broken.

6.6 A Note

We have assumed that original $NPF \leq 2K-2$, and have supplemented pieces to make a new $NPF = 2K$. In fact, we can assume that original $NPF \leq 3K-2$, and supplement pieces to make a new $NPF = 3K$. In this case, we can still break the instance, but our attack will be a little bit more complicated.

7 Cryptanalysis of Two Simple Revisions of GGH Map

7.1 The First Simple Revision of GGH Map and Corresponding MKE

The first simple revision of GGH map is described as follows. All parameters of GGH map are reserved, except that we change encoding parameter y into encoding parameters $\{y^{(i)}, i = 1, 2\}$, and accordingly we change Level-K zero-testing parameter p_{zt} into Level-K zero-testing parameters $\{p_{zt}^{(i)}, i = 1, 2\}$. Our encoding parameters are $\{y^{(i)}, i = 1, 2\}$, where $y^{(i)} = (y^{(0,i)} + a^{(i)}g)z^{-1}(\text{mod } q)$, $\{y^{(0,i)}, a^{(i)}, i = 1, 2\}$ are very small and are kept secret. We can see that $\{y^{(i)}, i = 1, 2\}$ are encodings of secret elements $\{y^{(0,i)}, i = 1, 2\}$, rather than encodings of 1. Accordingly, our level-K zero-testing parameters are$\{p_{zt}^{(i)}, i = 1, 2\}$, where $p_{zt}^{(i)} = hy^{(0,i)}z^K g^{-1}(\text{mod } q)$.

Suppose a user has a secret $(v^{(1)}, v^{(2)}) \in R^2$, where $v^{(1)}$ and $v^{(2)}$ are short elements. He/she secretly samples short elements $\{u^{(i)} \in R, i = 1, 2\}$. He/she computes noisy encoding $V = (v^{(1)}y^{(1)} + v^{(2)}y^{(2)}) + (u^{(1)}x^{(1)} + u^{(2)}x^{(2)})(\text{mod } q)$. He/she publishes V. Then, the first revision of GGH map includes K, $\{y^{(i)}, i = 1, 2\}$, $\{x^{(i)}, i = 1, 2\}$, $\{p_{zt}^{(i)}, i = 1, 2\}$, and all noisy encoding V for all users. To guarantee our attack work, we assume that 2^K is polynomially large.

Suppose that $K+1$ users want to generate KEY, a commonly shared key by public discussion. To do so, each user k generates his/her secret $(v^{(k,1)}, v^{(k,2)})$, and publishes the noisy encoding $V^{(k)}$, $k = 1, \ldots, K+1$. Then, each user can use his/her secret and other users' noisy encodings to compute KEY, the commonly shared key. For example, user k_0 first computes $(v^{(k_0,1)}p_{zt}^{(1)} + v^{(k_0,2)}p_{zt}^{(2)}) \prod_{k\neq k_0} V^{(k)} (\mathrm{mod}\ q)$, then takes KEY as its high-order bits. It is easy to see that

$$(v^{(k_0,1)}p_{zt}^{(1)} + v^{(k_0,2)}p_{zt}^{(2)}) \prod_{k\neq k_0} V^{(k)} (\mathrm{mod}\ q) = (A + B^{(k_0)}) (\mathrm{mod}\ q),$$

such that

$$A = hg^{-1} \sum_{(j_1,\ldots,j_{K+1})\in\{1,2\}^{K+1}} v^{(K+1,j_{K+1})} y^{(0,j_{K+1})} \prod_{k=1}^{K} v^{(k,j_k)} (y^{(0,j_k)} + a^{(j_k)} g) (\mathrm{mod}\ q),$$

which has no relation with user k_0; $B^{(k_0)}$ is the sum of several terms which are somewhat small. If related parameters are small enough, KEY is high-order bits of $A (\mathrm{mod}\ q)$.

7.2 Generating "Equivalent Secret"

For the secret $(v^{(1)}, v^{(2)}) \in R^2$, we construct an "equivalent secret $(v'^{(1)}, v'^{(2)}) \in R^2$", such that

$$\big(v^{(1)}(y^{(0,1)}+a^{(1)}g)+v^{(2)}(y^{(0,2)}+a^{(2)}g)\big)-\big(v'^{(1)}(y^{(0,1)}+a^{(1)}g)+v'^{(2)}(y^{(0,2)}+a^{(2)}g)\big)$$

is a multiple of g. An equivalent requirement is that $(v^{(1)}y^{(0,1)} + v^{(2)}y^{(0,2)}) - (v'^{(1)}y^{(0,1)} + v'^{(2)}y^{(0,2)})$ is a multiple of g. That is enough, and we do not need $(v'^{(1)}, v'^{(2)})$ small. Take V, the noisy encoding of $(v^{(1)}, v^{(2)})$, we compute special element

$$\begin{aligned} W^* = V(y^{(1)})^{K-2}x^{(1)}p_{zt}^{(1)} (\mathrm{mod}\ q) = hy^{(0,1)} \big[& v^{(1)}(y^{(0,1)} + a^{(1)}g)^{K-1}b^{(1)} \\ & + v^{(2)}(y^{(0,2)} + a^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} \\ & + u^{(1)}(b^{(1)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} \\ & + u^{(2)}(b^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} \big]. \end{aligned}$$

Notice that

(1) Right side of this equation has no operation "mod q", therefore W^* is somewhat small.

(2) Four vectors $hy^{(0,1)}(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)}$, $hy^{(0,1)}(y^{(0,2)}+a^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}$, $hy^{(0,1)}(b^{(1)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)}$ and $hy^{(0,1)}(b^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)}$ can be obtained.

Now we start to find $(v'^{(1)}, v'^{(2)})$. First, compute $W^*(\mathrm{mod}\ hy^{(0,1)}(y^{(0,1)} + a^{(1)}g)^{K-1}b^{(1)})$. Second, compute $\{v'^{(2)}, u'^{(1)}, u'^{(2)}\}$ such that

$$\begin{aligned} W^*(\mathrm{mod}\ \ & h\, y^{(0,1)}(y^{(0,1)} + a^{(1)}g)^{K-1}b^{(1)}) = \\ & h\, y^{(0,1)}\big[v'^{(2)}(y^{(0,2)} + a^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} + \\ & u'^{(1)}(b^{(1)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} + \\ & u'^{(2)}(b^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)}\big](\mathrm{mod}\ hy^{(0,1)}(y^{(0,1)} + a^{(1)}g)^{K-1}b^{(1)}). \end{aligned}$$

Solving this modular equation is quite easy algebra, as shown in Appendix A. Solutions are not unique, therefore $\{v'^{(2)}, u'^{(1)}, u'^{(2)}\} \neq \{v^{(2)}, u^{(1)}, u^{(2)}\}$. Third, compute $v'^{(1)}$ such that

$$\begin{aligned} W^* = hy^{(0,1)}\big[& v'^{(1)}(y^{(0,1)} + a^{(1)}g)^{K-1}b^{(1)} \\ & + v'^{(2)}(y^{(0,2)} + a^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} \\ & + u'^{(1)}(b^{(1)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)} \\ & + u'^{(2)}(b^{(2)}g)(y^{(0,1)} + a^{(1)}g)^{K-2}b^{(1)}\big], \end{aligned}$$

which is another version of easy algebra. Finally, we obtain $(v'^{(1)}, v'^{(2)})$, and can easily check that $(v^{(1)}(y^{(0,1)} + a^{(1)}g) + v^{(2)}(y^{(0,2)} + a^{(2)}g)) - (v'^{(1)}(y^{(0,1)} + a^{(1)}g) + v'^{(2)}(y^{(0,2)} + a^{(2)}g))$ is a multiple of g, although $v'^{(1)}$ and $v'^{(2)}$ are not short vectors.

7.3 Generalization of Modified Encoding/Zero-testing: Our Attack on MKE

Suppose $K+1$ users hide $(v^{(k,1)}, v^{(k,2)})$ and publish $V^{(k)}, k = 1, \ldots, K+1$, and for each user k we have obtained an equivalent secret $(v'^{(k,1)}, v'^{(k,2)})$. For each "$K+1$-dimensional boolean vector" $(j_1, \ldots, j_{K+1}) \in \{1,2\}^{K+1}$, we denote two products

$$v^{(j_1,\ldots,j_{K+1})} = \prod_{k=1}^{K+1} v^{(k,j_k)},$$

$$v'^{(j_1,\ldots,j_{K+1})} = \prod_{k=1}^{K+1} v'^{(k,j_k)}.$$

$v^{(j_1,\ldots,j_{K+1})}$ is clearly smaller than "somewhat small", because it does not include h. $v'^{(j_1,\ldots,j_{K+1})}$ is not a short vector. $v^{(j_1,\ldots,j_{K+1})}$ cannot be obtained, while $v'^{(j_1,\ldots,j_{K+1})}$ can. Suppose former K entries $\{j_1, \ldots, j_K\}$ include N_1 1s and N_2 2s, $N_1 + N_2 = K$. We denote the supporter $s^{(j_1,\ldots,j_{K+1})}$ as follows.

$$s^{(j_1,\ldots,j_{K+1})} = hy^{(0,j_{K+1})}(y^{(0,1)} + a^{(1)}g)^{N_1-1}(y^{(0,2)} + a^{(2)}g)^{N_2}b^{(1)} \quad \text{for } N_1 \geq N_2,$$

$$s^{(j_1,\ldots,j_{K+1})} = hy^{(0,j_{K+1})}(y^{(0,1)} + a^{(1)}g)^{N_1}(y^{(0,2)} + a^{(2)}g)^{N_2-1}b^{(1)} \quad \text{for } N_1 < N_2.$$

$s^{(j_1,\ldots,j_{K+1})}$ can be obtained. If $N_1 \geq N_2$, $s^{(j_1,\ldots,j_{K+1})} = p_{zt}^{(j_{K+1})}(y^{(1)})^{N_1-1}(y^{(2)})^{N_2}x^{(1)} (\mathrm{mod}\ q)$, and if $N_1 < N_2$, $s^{(j_1,\ldots,j_{K+1})} = p_{zt}^{(j_{K+1})}(y^{(1)})^{N_1}(y^{(2)})^{N_2-1}x^{(1)} (\mathrm{mod}\ q)$. $s^{(j_1,\ldots,j_{K+1})}$ is somewhat small. Then, we denote

$$V^{(N_1 \geq N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 \geq N_2} v^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V^{(N_1 < N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 < N_2} v^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V'^{(N_1 \geq N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 \geq N_2} v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V'^{(N_1 < N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 < N_2} v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})}.$$

$V^{(N_1 \geq N_2)}$ and $V^{(N_1 < N_2)}$ are somewhat small, while $V'^{(N_1 \geq N_2)}$ and $V'^{(N_1 < N_2)}$ are not short vectors. $V^{(N_1 \geq N_2)}$ and $V^{(N_1 < N_2)}$ cannot be obtained, while $V'^{(N_1 \geq N_2)}$ and $V'^{(N_1 < N_2)}$ can be obtained, because $v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})}$ can be obtained for each $(j_1, \ldots, j_{K+1}) \in \{1, 2\}^{K+1}$, and 2^K is polynomially large. Another fact is that ξ^* is a multiple of $b^{(1)}g$, where

$$\xi^* = (y^{(0,1)} + a^{(1)}g)(V'^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}) + (y^{(0,2)} + a^{(2)}g)(V'^{(N_1 < N_2)} - V^{(N_1 < N_2)}).$$

There are two reasons: (1) By considering the definitions of equivalent secrets, we know that ξ^* is a multiple of g. (2) By considering the definition of $s^{(j_1,\ldots,j_{K+1})}$, we know that ξ^* is a multiple of $b^{(1)}$. Here we use a small assumption that $b^{(1)}$ and g are co-prime. Notice that ξ^* is not a short vector, and that ξ^* cannot be obtained. Then, we compute a tool for the modular operations,

$$M = hy^{(0,1)}(b^{(1)})^K g^{K-1} = p_{zt}^{(1)}(x^{(1)})^K (\mathrm{mod}\ q).$$

For the same reason, M is somewhat small. Then, we compute the modular operations

$$V''^{(N_1 \geq N_2)} = V'^{(N_1 \geq N_2)} (\mathrm{mod}\ M),$$

$$V''^{(N_1 < N_2)} = V'^{(N_1 < N_2)} (\mathrm{mod}\ M).$$

Both $V''^{(N_1 \geq N_2)}$ and $V''^{(N_1 < N_2)}$ are somewhat small. Therefore, both $V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}$ and $V''^{(N_1 < N_2)} - V^{(N_1 < N_2)}$ are somewhat small. Therefore, both $(y^{(0,1)} + a^{(1)}g)(V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)})$ and $(y^{(0,2)} + a^{(2)}g)(V''^{(N_1 < N_2)} - V^{(N_1 < N_2)})$ are somewhat small. Therefore,

$$\xi^{**} = (y^{(0,1)} + a^{(1)}g)(V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}) + (y^{(0,2)} + a^{(2)}g)(V''^{(N_1 < N_2)} - V^{(N_1 < N_2)})$$

is somewhat small. On the other hand, ξ^{**} is a multiple of $b^{(1)}g$, because ξ^{*} is a multiple of $b^{(1)}g$. Therefore, $\xi^{**}/(b^{(1)}g)$ is somewhat small. Finally,

$$\begin{aligned}\frac{\xi^{**}}{(b^{(1)}g)} &= \xi^{**}(b^{(1)}g)^{-1}(\text{mod } q)\\ &= \Big[\Big((y^{(0,1)}+a^{(1)}g)V''^{(N_1\geq N_2)}+(y^{(0,2)}+a^{(2)}g)V''^{(N_1<N_2)}\Big)(b^{(1)}g)^{-1}-A\Big](\text{mod } q),\end{aligned}$$

which means that KEY is high-order bits of

$$\Big[\Big((y^{(0,1)}+a^{(1)}g)V''^{(N_1\geq N_2)}+(y^{(0,2)}+a^{(2)}g)V''^{(N_1<N_2)}\Big)(b^{(1)}g)^{-1}\Big](\text{mod } q),$$

which can be obtained, because $(y^{(0,1)}+a^{(1)}g)(b^{(1)}g)^{-1}(\text{mod } q)$ and $(y^{(0,2)}+a^{(2)}g)(b^{(1)}g)^{-1}(\text{mod } q)$ can be obtained.

7.4 The Second Simple Revision of GGH Map and Its Cryptanalysis

The second simple revision of GGH map is described as follows. All parameters of the first simple revision are reserved, except that we change K-order zero-testing parameters $\{p_{zt}^{(i)}=hy^{(0,i)}z^Kg^{-1}(\text{mod } q), i=1,2\}$ into $\{p_{zt}^{(i)}=(y^{(0,i)}+h^{(i)}g)z^Kg^{-1}(\text{mod } q), i=1,2\}$, where both $h^{(1)}$ and $h^{(2)}$ are somewhat small sampled with standard deviation $\sqrt{q}$. MKE is just the same procedure as the first simple revision, except for the different $\{p_{zt}^{(i)}, i=1,2\}$. Such a structure can be taken as a simplified version of Gu map-1 [25]. Our cryptanalysis obtains the same result: MKE can be broken under the the assumption that 2^K is polynomially large. The deduction procedure is almost same, and we present it in Appendix D.

8 Some Considerations and Remaining Questions

There are many different variants of the GGH construction that one can consider, below we briefly discuss one of them. The variant which seems to defeat our attacks is using non-commutative operations (e.g., using matrices). However this greatly reduces the usability of this construction, for example the WE construction based on X3C requires commutativity. Other variants are under our study.

Trying to find extensions of these attacks and their limitations remains an interesting research direction. For example, we do not know whether the two simple revisions that we analyzed above can be used to construct a secure WE scheme based on X3C. It will also be very interesting to find a way to use our attacks against GGH-based obfuscation schemes.

Acknowledgments. We thank all the reviewers and editors for their valuable comments and works. We also appreciate Martin R. Albrecht for his implementation of our attack [27]. We are very grateful for help and suggestions from the

authors of GGH map [2] and authors of the instance of WE based on the hardness of X3C problem [3]. We are very grateful to professor Dong Pyo Chi from UNIST, Korea, for pointing out mistakes in our work. This work was supported in part by the Natural Science Foundation of China under Grant 61173151 and 61472309.

References

1. Boneh, D., Silverberg, A.: Applications of Multilinear Forms to Cryptography. Contemporary Mathematics. 324: 71–90 (2003)
2. Garg, S., Gentry, C., Halevi, S.: Candidate Multilinear Maps from Ideal Lattices. In: Johansson, T., Nguyen, P.Q. (ed.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 181–184. Springer, Heidelberg (2013)
3. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness Encryption and its Applications. In: STOC (2013)
4. Gentry, C., Lewko, A., Waters, B.: Witness Encryption from Instance Independent Assumptions. In: Garay, J.A., Gennaro, R. (ed.) CRYPTO 2014. LNCS, vol. 8616, pp. 426–443. Springer, Heidelberg (2014)
5. Arita, S., Handa, S.: Two Applications of Multilinear Maps: Group Key Exchange and Witness Encryption. In: Proceedings of the 2nd ACM workshop on ASIA public-key cryptography(ASIAPKC '14). ACM, New York, NY, USA, pp. 13–22 (2014)
6. Bellare, M., Hoang, V.T.: Adaptive Witness Encryption and Asymmetric Password-Based Cryptography. Cryptology ePrint Archive, Report 2013/704 (2013)
7. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to Run Turing Machines on Encrypted Data. In: Canetti, R., Garay, J.A. (ed.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 536–553. Springer, Heidelberg (2013)
8. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate Indistinguishability Obfuscation and Functional Encryption for all Circuits. In: FOCS (2013)
9. Garg, S., Gentry, C., Halevi, S., Wichs, D.: On the Implausibility of Differing-Inputs Obfuscation and Extractable Witness Encryption with Auxiliary Input. In: Garay, J.A., Gennaro, R. (ed.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 518–535. Springer, Heidelberg (2014)
10. Boyle, E., Chung, K.-M., Pass, R.: On Extractability (a.k.a. Differing-Input) Obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 52–73. Springer, Heidelberg (2014)
11. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-Based Encryption for Circuits from Multilinear Maps. In: Canetti, R., Garay, J.A. (ed.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 479–499. Springer, Heidelberg (2013)
12. Langlois, A., Stehlé, D., Steinfeld, R.: GGHLiteMore Efficient Multilinear Maps from Ideal Lattices. In: Nguyen, P.Q., Oswald, E. (ed.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 239–256. Springer, Heidelberg (2014)
13. Coron, J.-S., Lenpoint, T., Tibouchi, M.: Practical Multilinear Maps over the Integers. In: Canetti, R., Garay, J.A. (ed.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 476–493. Springer, Heidelberg (2013)
14. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé. D: Cryptanalysis of the Multilinear Map over the Integers. In: Oswald, E., Fischlin, M. (ed.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 3–12. Springer, Heidelberg (2015)

15. Gentry, C., Halevi, S., Maji, H.K., Sahai, A.: Zeroizing without Zeroes: Cryptanalyzing Multilinear Maps without Encodings of Zero. Cryptology ePrint Archive, Report 2014/929 (2014)
16. Boneh, D., Wu, D.J., Zimmerman, J.: Immunizing Multilinear Maps Against Zeroizing Attacks. Cryptology ePrint Archive, Report 2014/930 (2014)
17. Coron, J.-S., Lepoint, T., Tibouchi, M.: Cryptanalysis of Two Candidate Fixes of Multilinear Maps over the Integers. Cryptology ePrint Archive, Report 2014/975 (2014)
18. Gentry, C., Gorbunov, S., Halevi, S.: Graph-Induced Multilinear Maps from Lattices. In: Dodis, Y. and Nielsen, J.B. (ed.) TCC 2015, Part II, LNCS, vol. 9015, pp. 498–527. Springer, Heidelberg (2015)
19. Coron, J.-S., Gentry, C., Halevi, S., Lepoint, T., Maji H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing Without Low-level Zeroes: New MMAP Attacks and their Limitations. In: Gennaro, R., Robshaw, M. (ed.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 247–266. Springer, Heidelberg (2015)
20. Coron, J.-S., Lepoint, T., Tibouchi, M.: New Multilinear Maps over the Integers. In: Gennaro, R., Robshaw, M. (ed.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 267–286. Springer, Heidelberg (2015)
21. Cheon, J.H., Han, K., Lee, C., Ryu, H.: Cryptanalysis of the New CLT Multilinear Maps. Cryptology ePrint Archive, Report 2015/934 (2015)
22. Minaud, B., Fouque, P.-A.: Cryptanalysis of the New Multilinear Map over the Integers. Cryptology ePrint Archive, Report 2015/941 (2015)
23. Babai, L.: On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica 6(1): 1–13 (1986)
24. Goldreich, O.: Computational Complexity: a Conceptual Perspective. Cambridge University Press, New York, NY, USA, 1 edition (2008)
25. Gu, C.: Multilinear Maps Using Ideal Lattices without Encodings of Zero. Cryptology ePrint Archive, Report 2015/023 (2015)
26. Nguyen, P.Q., Regev, O.: Learning a Parallel Piped: Cryptanalysis of GGH and NTRU Signatures. Journal of Cryptology. 22(2), 139–160 (2009)
27. Albrecht, M.R.: Sage Code for GGH Cryptanalysis by Hu and Jia. Available at https://martinralbrecht.wordpress.com/2015/04/13/sage-code-for-ggh-cryptanalysis-by-hu-and-jia/

Appendix

A

Suppose $W(\text{mod } Y) = W'''Y$, $X^{(1)}(\text{mod } Y) = X'^{(1)}Y$, and $X^{(2)}(\text{mod } Y) = X'^{(2)}Y$. We want to obtain a solution $u'^{(i)} \in R$, $i = 1, 2$, such that $W'''Y = (u'^{(1)}X'^{(1)} + u'^{(2)}X'^{(2)})Y(\text{mod } Y)$. First, the equation has solution, because $\{u^{(i)} \in R, i = 1, 2\}$ is a solution. Second, the equation can be modified as an equivalent equation $W''' = (u'^{(1)}X'^{(1)} + u'^{(2)}X'^{(2)})(\text{mod } 1)$. Third, take each entry of W''', $X'^{(1)}$, and $X'^{(2)}$ as the form of reduced fraction, and take LCM as the least common multiple of all denominators, then the equation can be modified as an equivalent equation, which is a linear equation modular LCM:

$$(LCM)W''' = (u'^{(1)}((LCM)X'^{(1)}) + u'^{(2)}((LCM)X'^{(2)}))(\text{mod } (LCM)).$$

B

Arbitrarily take a subset $\{i_1, i_2, i_3\}$ which is not a piece. We will compute $P(\{i_1, i_2, i_3\}$ is not a combined piece). First, we take a random experiment: randomly choosing 3 subsets $\{j_1, j_2, j_3\}$, $\{k_1, k_2, k_3\}$, $\{l_1, l_2, l_3\}$ from $\{1, 2, \ldots, 3K\}$. Then, the probability of such event:

$$\{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} \supset \{i_1, i_2, i_3\},$$

$$\{l_1, l_2, l_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{i_1, i_2, i_3\},$$

is

$$\frac{C_{3K}^3 C_6^3}{(C_{3K}^3)^3} \approx \frac{1}{K^6}.$$

Second, from $O(K^2)$ pieces we generate all 3-tuples of 3 pieces $\{\{j_1, j_2, j_3\}, \{k_1, k_2, k_3\}, \{l_1, l_2, l_3\}\}$. We know there are $O(K^2)(O(K^2) - 1)(O(K^2) - 2)$ 3-tuples. Then, the probability of such event: there is no a 3-tuples $\{\{j_1, j_2, j_3\}, \{k_1, k_2, k_3\}, \{l_1, l_2, l_3\}\}$, such that

$$\{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} \supset \{i_1, i_2, i_3\},$$

$$\{l_1, l_2, l_3\} = \{j_1, j_2, j_3\} \cup \{k_1, k_2, k_3\} - \{i_1, i_2, i_3\},$$

is about

$$\Big(1 - \frac{1}{K^6}\Big)^{O(K^2)(O(K^2)-1)(O(K^2)-2)} \approx exp\Big\{ - \frac{(O(K^2))^3}{K^6}\Big\}.$$

C

We need to obtain Hermite normal form $G = \begin{bmatrix} G_0 & & & \\ G_1 & 1 & & \\ \vdots & & \ddots & \\ G_{n-1} & & & 1 \end{bmatrix}$, where each row of G is an element of $\langle g \rangle$, G_0 is the absolute value of the determinant of the matrix $\begin{bmatrix} g_0 & g_1 & \cdots & g_{n-1} \\ -g_{n-1} & g_0 & \cdots & g_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ -g_1 & -g_2 & \cdots & g_0 \end{bmatrix}$, and $G_i (\text{mod } G_0) = G_i$ for $i = 1, \ldots, n-1$.

For a principal ideal $\langle g' \rangle$, we call the determinant of $\begin{bmatrix} g'_0 & g'_1 & \cdots & g'_{n-1} \\ -g'_{n-1} & g'_0 & \cdots & g'_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ -g'_1 & -g'_2 & \cdots & g'_0 \end{bmatrix}$ corresponding determinant of $\langle g' \rangle$. We use the definition of parallel piped [26]. For a vector $\alpha \in R$, we call the set $PP(\alpha) = \{z \in R : z(\text{mod } \alpha) = z\}$ parallel piped of α.

Two facts. We have $\{Y, X^{(i)}, i = 1, 2\}$, therefore we can obtain hermite normal forms of the principal ideals $\{\langle Y\rangle, \langle X^{(i)}\rangle, i = 1, 2\}$.

Suppose Hermite normal form of the principal ideal $\langle g'\rangle$ is $\begin{bmatrix} G'_0 & & & \\ G'_1 & 1 & & \\ \vdots & & \ddots & \\ G'_{n-1} & & & 1 \end{bmatrix}$, $g \in R$ is a factor of g', and absolute value of corresponding determinant of $\langle g\rangle$ is G_0. Then, Hermite normal form of the principal ideal $\langle g\rangle$ is $\begin{bmatrix} G_0 & & & \\ G'_1 (\mathrm{mod}\ G_0) & 1 & & \\ \vdots & & \ddots & \\ G'_{n-1} (\mathrm{mod}\ G_0) & & & 1 \end{bmatrix}$.

Computing Hermite normal form of $\langle h(1+ag)^{K-2}b^{(1)}\rangle$. We take a trivial assumption that $1 + ag$ and $b^{(1)}g$ are co-prime.

Step 1. By using $\{Y, (-Y_{n-1}, Y_0, \ldots, Y_{n-2}), \ldots, (-Y_1, \ldots, -Y_{n-1}, Y_0)\}$ as the basis, Gaussian sample Z, with sufficiently large deviation.

Step 2. Compute $Z' = Z(\mathrm{mod}\ X^{(1)})$. Then, Z' is uniformly distributed over the intersection area $\langle h(1+ag)^{K-2}b^{(1)}\rangle \cap PP(X^{(1)})$. Algebra and Gaussian sampling theory have proven this result.

Step 3. Compute absolute value of corresponding determinant of $\langle Z'\rangle$.

Step 4. Repeat Step 1∼3 polynomially many times, so that we obtain polynomially many absolute values of corresponding determinant.

Step 5. Compute the greatest common divisor of these polynomially many absolute values. Then, the greatest common divisor should be absolute value of corresponding determinant of $\langle h(1 + ag)^{K-2}b^{(1)}\rangle$. By considering a fact stated in last subsection, we obtain Hermite normal form of $\langle h(1 + ag)^{K-2}b^{(1)}\rangle$.

Computing Hermite normal form of $\langle h(1 + ag)^{K-2}b^{(1)}g\rangle$. We take a trivial assumption that $b^{(1)}$ and $b^{(2)}$ are co-prime. The procedure is similar to last subsection.

Step 1. By using $\{X^{(2)}, (-X^{(2)}_{n-1}, X^{(2)}_0, \ldots, X^{(2)}_{n-2}), \ldots, (-X^{(2)}_1, \ldots, -X^{(2)}_{n-1}, X^{(2)}_0)\}$ as the basis, Gaussian sample Z, with sufficiently large deviation.

Step 2. Compute $Z' = Z(\mathrm{mod}\ X^{(1)})$. Then, Z' is uniformly distributed over the intersection area $\langle h(1 + ag)^{K-2}b^{(1)}g\rangle \cap PP(X^{(1)})$.

Step 3. Compute absolute value of corresponding determinant of $\langle Z'\rangle$.

Step 4. Repeat Step 1∼3 polynomially many times, so that we obtain polynomially many absolute values of corresponding determinant.

Step 5. Compute the greatest common divisor of these polynomially many absolute values. Then, the greatest common divisor should be absolute value of corresponding determinant of $\langle h(1+ag)^{K-2}b^{(1)}g\rangle$, therefore, we obtain Hermite normal form of $\langle h(1 + ag)^{K-2}b^{(1)}g\rangle$.

Obtaining Hermite normal form of $\langle g\rangle$. Divide absolute value of corresponding determinant of $\langle h(1+ag)^{K-2}b^{(1)}g\rangle$ by absolute value of corresponding determinant of $\langle h(1+ag)^{K-2}b^{(1)}\rangle$. Then, we obtain absolute value of corresponding determinant of $\langle g\rangle$, therefore we obtain Hermite normal form of $\langle g\rangle$.

D

Here we use several symbols which have been used for analyzing the first simple revision of GGH map. User k_0 first computes $(v^{(k_0,1)}p_{zt}^{(1)}+v^{(k_0,2)}p_{zt}^{(2)})\prod_{k\neq k_0}V^{(k)}(\mathrm{mod}\ q)$, then takes KEY as its high-order bits. It is easy to see that

$$(v^{(k_0,1)}p_{zt}^{(1)}+v^{(k_0,2)}p_{zt}^{(2)})\prod_{k\neq k_0}V^{(k)}(\mathrm{mod}\ q)=(A+B^{(k_0)})(\mathrm{mod}\ q),$$

such that

$$A=g^{-1}\sum_{(j_1,\ldots,j_{K+1})\in\{1,2\}^{K+1}}v^{(K+1,j_{K+1})}(y^{(0,j_{K+1})}+h^{(j_{K+1})}g)\prod_{k=1}^{K}v^{(k,j_k)}(y^{(0,j_k)}+a^{(j_k)}g)(\mathrm{mod}\ q),$$

which has no relation with user k_0; $B^{(k_0)}$ is the sum of several terms which are somewhat small. If related parameters are small enough, KEY is high-order bits of $A(\mathrm{mod}\ q)$.

Generating "equivalent secret". For the secret $(v^{(1)},v^{(2)})\in R^2$, we construct an "equivalent secret $(v'^{(1)},v'^{(2)})\in R^2$", such that

$$\big(v^{(1)}(y^{(0,1)}+a^{(1)}g)+v^{(2)}(y^{(0,2)}+a^{(2)}g)\big)-\big(v'^{(1)}(y^{(0,1)}+a^{(1)}g)+v'^{(2)}(y^{(0,2)}+a^{(2)}g)\big)$$

is a multiple of g. One equivalent requirement is that $(v^{(1)}y^{(0,1)}+v^{(2)}y^{(0,2)})-(v'^{(1)}y^{(0,1)}+v'^{(2)}y^{(0,2)})$ is a multiple of g. Another equivalent requirement is that

$$\big(v^{(1)}(y^{(0,1)}+h^{(1)}g)+v^{(2)}(y^{(0,2)}+h^{(2)}g)\big)-\big(v'^{(1)}(y^{(0,1)}+h^{(1)}g)+v'^{(2)}(y^{(0,2)}+h^{(2)}g)\big)$$

is a multiple of g. That is enough, and we do not need $(v'^{(1)},v'^{(2)})$ small. Take V, the noisy encoding of $(v^{(1)},v^{(2)})$, we compute special element

$$\begin{aligned}W^*=V(y^{(1)})^{K-2}x^{(1)}p_{zt}^{(1)}(\mathrm{mod}\ q)=(y^{(0,1)}+h^{(1)}g)\big[&v^{(1)}(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)}\\&+v^{(2)}(y^{(0,2)}+a^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}\\&+u^{(1)}(b^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}\\&+u^{(2)}(b^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}\big].\end{aligned}$$

Notice that

(1) Right side of this equation has no operation "mod q", therefore W^* is somewhat small.

(2) Four vectors $(y^{(0,1)}+h^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)}$, $(y^{(0,1)}+h^{(1)}g)(y^{(0,2)}+a^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}$, $(y^{(0,1)}+h^{(1)}g)(b^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}$ and $hy^{(0,1)}(b^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}$ can be obtained.

Now we start to find $(v'^{(1)}, v'^{(2)})$. First, compute $W^*(\mathrm{mod}\ (y^{(0,1)}+h^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)})$. Second, compute $\{v'^{(2)}, u'^{(1)}, u'^{(2)}\}$ such that

$$\begin{aligned} W^*(\mathrm{mod}\quad & (y^{(0,1)}+h^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)}) = \\ & (y^{(0,1)}+h^{(1)}g)\big[v'^{(2)}(y^{(0,2)}+a^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)} + \\ & u'^{(1)}(b^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)} + \\ & u'^{(2)}(b^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}\big](\mathrm{mod}\ (y^{(0,1)}+h^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)}). \end{aligned}$$

Solving this modular equation is quite easy algebra, as in Appendix A. Solutions are not unique, therefore $\{v'^{(2)}, u'^{(1)}, u'^{(2)}\} \neq \{v^{(2)}, u^{(1)}, u^{(2)}\}$. Third, compute $v'^{(1)}$ such that

$$\begin{aligned} W^* = (y^{(0,1)}+h^{(1)}g)\big[& v'^{(1)}(y^{(0,1)}+a^{(1)}g)^{K-1}b^{(1)} \\ & + v'^{(2)}(y^{(0,2)}+a^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)} \\ & + u'^{(1)}(b^{(1)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)} \\ & + u'^{(2)}(b^{(2)}g)(y^{(0,1)}+a^{(1)}g)^{K-2}b^{(1)}\big], \end{aligned}$$

which is another easy algebra. Finally, we obtain $(v'^{(1)}, v'^{(2)})$, and can easily check that $(v^{(1)}(y^{(0,1)}+a^{(1)}g)+v^{(2)}(y^{(0,2)}+a^{(2)}g)) - (v'^{(1)}(y^{(0,1)}+a^{(1)}g)+v'^{(2)}(y^{(0,2)}+a^{(2)}g))$ is a multiple of g, although $v'^{(1)}$ and $v'^{(2)}$ are not short vectors.

Generalization of modified encoding/zero-testing: our attack on MKE. Suppose $K+1$ users hide $(v^{(k,1)}, v^{(k,2)})$ and publish $V^{(k)}, k = 1, \ldots, K+1$, and for each user k we have obtained equivalent secret $(v'^{(k,1)}, v'^{(k,2)})$. For each "$K+1$-dimensional boolean vector" $(j_1, \ldots, j_{K+1}) \in \{1,2\}^{K+1}$, we denote two products

$$v^{(j_1,\ldots,j_{K+1})} = \prod_{k=1}^{K+1} v^{(k,j_k)},$$

$$v'^{(j_1,\ldots,j_{K+1})} = \prod_{k=1}^{K+1} v'^{(k,j_k)}.$$

$v^{(j_1,\ldots,j_{K+1})}$ is clearly smaller than "somewhat small", because it does not contain $h^{(1)}$ and $h^{(2)}$. $v'^{(j_1,\ldots,j_{K+1})}$ is not a short vector. $v^{(j_1,\ldots,j_{K+1})}$ cannot be obtained, while $v'^{(j_1,\ldots,j_{K+1})}$ can. Suppose former K entries $\{j_1, \ldots, j_K\}$ include N_1 1s and N_2 2s, $N_1+N_2=K$. We denote the supporter $s^{(j_1,\ldots,j_{K+1})}$ as follows.

$$s^{(j_1,\ldots,j_{K+1})} = (y^{(0,j_{K+1})}+h^{(j_{K+1})}g)(y^{(0,1)}+a^{(1)}g)^{N_1-1}(y^{(0,2)}+a^{(2)}g)^{N_2}b^{(1)} \quad \text{for } N_1 \geq N_2,$$

$$s^{(j_1,\ldots,j_{K+1})} = (y^{(0,j_{K+1})}+h^{(j_{K+1})}g)(y^{(0,1)}+a^{(1)}g)^{N_1}(y^{(0,2)}+a^{(2)}g)^{N_2-1}b^{(1)} \quad \text{for } N_1 < N_2.$$

$s^{(j_1,\ldots,j_{K+1})}$ can be obtained. If $N_1 \geq N_2$, $s^{(j_1,\ldots,j_{K+1})} = p_{zt}^{(j_{K+1})}(y^{(1)})^{N_1-1}(y^{(2)})^{N_2}x^{(1)} (\mathrm{mod}\ q)$, and if $N_1 < N_2$, $s^{(j_1,\ldots,j_{K+1})} = p_{zt}^{(j_{K+1})}(y^{(1)})^{N_1}(y^{(2)})^{N_2-1}x^{(1)} (\mathrm{mod}\ q)$. $s^{(j_1,\ldots,j_{K+1})}$

is somewhat small. Then, we denote

$$V^{(N_1 \geq N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 \geq N_2} v^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V^{(N_1 < N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 < N_2} v^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V'^{(N_1 \geq N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 \geq N_2} v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})},$$

$$V'^{(N_1 < N_2)} = \sum_{j_{K+1}=1}^{2} \sum_{N_1 < N_2} v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})}.$$

$V^{(N_1 \geq N_2)}$ and $V^{(N_1 < N_2)}$ are somewhat small, while $V'^{(N_1 \geq N_2)}$ and $V'^{(N_1 < N_2)}$ are not short vectors. $V^{(N_1 \geq N_2)}$ and $V^{(N_1 < N_2)}$ cannot be obtained, while $V'^{(N_1 \geq N_2)}$ and $V'^{(N_1 < N_2)}$ can be obtained, because $v'^{(j_1,\ldots,j_{K+1})} s^{(j_1,\ldots,j_{K+1})}$ can be obtained for each $(j_1, \ldots, j_{K+1}) \in \{1, 2\}^{K+1}$, and 2^K is polynomially large. Another fact is that ξ^* is a multiple of $b^{(1)}g$, where

$$\xi^* = (y^{(0,1)} + a^{(1)}g)(V'^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}) + (y^{(0,2)} + a^{(2)}g)(V'^{(N_1 < N_2)} - V^{(N_1 < N_2)}).$$

There are two reasons: (1) By considering the definitions of equivalent secrets, we know that ξ^* is a multiple of g. (2) By considering the definition of $s^{(j_1,\ldots,j_{K+1})}$, we know that ξ^* is a multiple of $b^{(1)}$. Here we use a small assumption that $b^{(1)}$ and g are co-prime. Notice that ξ^* is not a short vector, and that ξ^* cannot be obtained. Then, we compute a tool for modular operations,

$$M = (y^{(0,1)} + h^{(1)}g)(b^{(1)})^K g^{K-1} = p_{zt}^{(1)} (x^{(1)})^K (\text{mod } q).$$

For the same reason, M is somewhat small. Then, we compute the modular operations

$$V''^{(N_1 \geq N_2)} = V'^{(N_1 \geq N_2)} (\text{mod } M),$$
$$V''^{(N_1 < N_2)} = V'^{(N_1 < N_2)} (\text{mod } M).$$

Both $V''^{(N_1 \geq N_2)}$ and $V''^{(N_1 < N_2)}$ are somewhat small. Therefore, both $V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}$ and $V''^{(N_1 < N_2)} - V^{(N_1 < N_2)}$ are somewhat small. Therefore, both $(y^{(0,1)} + a^{(1)}g)(V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)})$ and $(y^{(0,2)} + a^{(2)}g)(V''^{(N_1 < N_2)} - V^{(N_1 < N_2)})$ are somewhat small. Therefore,

$$\xi^{**} = (y^{(0,1)} + a^{(1)}g)(V''^{(N_1 \geq N_2)} - V^{(N_1 \geq N_2)}) + (y^{(0,2)} + a^{(2)}g)(V''^{(N_1 < N_2)} - V^{(N_1 < N_2)})$$

is somewhat small. On the other hand, ξ^{**} is a multiple of $b^{(1)}g$, because ξ^* is a multiple of $b^{(1)}g$. Therefore, $\xi^{**}/(b^{(1)}g)$ is somewhat small. Finally,

$$\frac{\xi^{**}}{(b^{(1)}g)} = \xi^{**}(b^{(1)}g)^{-1} (\text{mod } q)$$

$$= \left[\left((y^{(0,1)} + a^{(1)}g)V''^{(N_1 \geq N_2)} + (y^{(0,2)} + a^{(2)}g)V''^{(N_1 < N_2)}\right)(b^{(1)}g)^{-1} - A\right] (\text{mod } q),$$

which means that KEY is high-order bits of

$$\left[\left((y^{(0,1)} + a^{(1)}g)V''^{(N_1 \geq N_2)} + (y^{(0,2)} + a^{(2)}g)V''^{(N_1 < N_2)}\right)(b^{(1)}g)^{-1}\right] (\text{mod } q).$$