Preventing Adaptive Key Recovery Attacks on the Gentry-Sahai-Waters Leveled Homomorphic Encryption Scheme*

Zengpeng Li[1,2**], Steven D. Galbraith[3], and Chunguang Ma[1,2]

[1] College of Computer Science and Technology, Harbin Engineering University, Harbin, China
[2] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[3] Department of Mathematics, The University of Auckland, Auckland, New Zealand.
{lizengpeng,machunguang}@hrbeu.edu.cn,
s.galbraith@auckland.ac.nz

Abstract. A major open problem is to protect leveled homomorphic encryption from adaptive attacks that allow an adversary to learn the private key. The only positive results in this area are by Loftus, May, Smart and Vercauteren. They use a notion of "valid ciphertexts" and obtain an IND-CCA1 scheme under a strong knowledge assumption, but they also show their scheme is not secure under a natural adaptive attack based on a "ciphertext validity oracle". However, due to recent cryptanalysis their scheme is no longer considered secure.
The main contribution of this paper is to explore a new approach to achieving this goal, which does not rely on a notion of "valid ciphertexts". The idea is to generate a "one-time" private key every time the decryption algorithm is run, so that even if an attacker can learn some bits of the one-time private key from each decryption query, this does not allow them to compute a valid private key.
This is the full version of the paper. The short version, which appeared in Provsec 2016, presented a variant of the Gentry-Sahai-Waters (GSW) levelled homomorphic encryption scheme. Damien Stehlé pointed out an attack on our variant of this scheme that had not been anticipated in the Provsec paper; we explain the attack in this full version. This version of the paper also contains a new "dual" version of the GSW scheme. We give an explanation of why the known attacks no longer break the system. It remains an open problem to develop a scheme for which one can prove IND-CCA1 security.

1 Introduction

It is well-known that access to a decryption oracle can lead to attacks on basic Regev [28] or Gentry-Peikert-Vaikuntanathan (GPV) [17] encryption, as well as various homomorphic encryption schemes [12,13,15,20,30]. These attacks allow an adversary to learn the private key, and so they are more serious than attacks that learn some information about messages.

* An early version of this paper appeared in Provsec2016. The current version has been completely revised and contains significant new material.
** Most work done while visiting The University of Auckland.

If homomorphic encryption is not required then there are general methods to obtain IND-CCA2 encryption from lattice problems (see [17], for example). However, these approaches are not compatible with homomorphic encryption. Hence, it is of major interest to obtain CCA1-secure variants of these schemes and this problem seems to be very difficult.

Loftus, May, Smart and Vercauteren [20] have considered the security of the private key of Gentry's homomorphic encryption scheme based on ideal lattices [16] (and some variants of it [29]) under adaptive attacks. They show that the private key can be determined if one has access to a decryption oracle. They give a variant of the Smart-Vercauteren cryptosystem [29] for which the private key seems to be secure even when a decryption oracle is present; this result is based on a notion of "valid ciphertext", which is checked by the decryption algorithm, and the security relies on a very strong knowledge assumption. Subsequently the computational assumption underlying the Smart-Vercauteren cryptosystem (the short principal ideal problem) has been broken [5,6,14], and so this scheme is no longer considered secure. Hence the problem of IND-CCA1 levelled homomorphic encryption remains an open problem.

Loftus et al emphasise the relevance of ciphertext validity attacks (CVA). This model allows an attacker to have access to an oracle that determines whether or not a ciphertext is valid. They show that it is possible for an adversary to decrypt a challenge ciphertext with the help of the CVA oracle (but at least the private key remains secure, and this is not a CCA1 attack but a CCA2 attack).

Loftus et al argue that CCA1 and CVA attacks on homomorphic encryption schemes are realistic in practice (they write in Section 6 of [20] that "Such an oracle can often be obtained in the real world by the attacker observing the behaviour of a party who is fed ciphertexts of the attacker's choosing"). For example, if a user is storing an encrypted database in the cloud and making queries to it, then an attacker could send ciphertexts of its choosing in response. If these ciphertexts are invalid then the user might re-send the same query until a valid ciphertext is received in response. Such a situation precisely gives a CVA oracle. Bleichenbacher's use of a CVA oracle to attack certain variants of RSA is well-known [7]. Hence, we believe that this issue is serious and that it is important to develop techniques to secure the private key of homomorphic encryption schemes.

In this paper we consider a different approach to the problem. Rather than relying on a notion of "valid ciphertexts", we avoid the risk of private key exposure by using "one-time" private keys. The idea is that, even if an attacker can learn some bits of the one-time private key from each decryption query, there should be no way for the attacker to combine the information from multiple decryption queries to compute an actual private key.

The idea of one-time secret keys can be implemented with many lattice-based cryptosystems, such as those based on LWE by Brakerski et al [8,10,11], but it only gives rise to a somewhat homomorphic scheme. We focus our attention on the Gentry-Sahai-Waters (GSW) scheme, since it can achieve leveled homomorphic encryption without any key switching; see Section 3 for a description of this scheme. This is important, since it is trivially impossible to achieve CCA1-security for any scheme that uses key-

switching or bootstrapping or any other method where the public key contains encryptions of secret information.

The short version of this paper, published at Provsec 2016, gives a variant of the GSW scheme that allows multiple secret keys. We present this scheme in Section 4. We give an argument using the left-over hash lemma that this scheme resists one of the standard adaptive attacks. However, Damien Stehlé pointed out an attack on that version of our scheme. We present this attack and give some analysis of it in Sections 3.3 and 5.2.

In Section 6 we present a "dual" version of the GSW scheme and show how this can be used with multiple secret keys. We explain how this scheme seems to avoid adaptive attacks. Unfortunately we are not able to prove IND-CCA1 security of this scheme. Indeed, proving IND-CCA1 security of homomorphic encryption is an extremely challenging unsolved problem. The nearest to a solution is the result of Loftus et al [20], which uses a very strong knowledge assumption and is now broken anyway.

To summarise, we have a general approach to protecting homomorphic encryption schemes of a certain type from adaptive attacks. Using the left-over hash lemma and an argument about projections of vector spaces we have a theoretical framework for understanding why a scheme resists adaptive attacks of a certain type. We hope that our ideas are a stepping-stone to a scheme with a proof of IND-CCA1 security.

The paper is organised as follows. Section 2 recalls some basic notions in the subject. Section 3 presents the Gentry-Sahai-Waters (GSW13) scheme. Section 4 presents our new scheme, while Section 7 explains why this scheme is resistant to the known adaptive attack. In the conclusions section we discuss whether our ideas might also be useful in the context of leakage resilience and side-channel protection.

2 Preliminaries

In this section we introduce some notations and recall the learning with errors problem. We also prove Theorem 2, which is an important tool for our main result.

2.1 Notation

We let $\mathbb{N}$ denote the natural numbers. For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \cdots, n\}$. For a real number $x \in \mathbb{R}$, we let $\lfloor x \rfloor$ denote the largest integer not greater than x, and $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ denote the integer closest to x, with ties broken upward. Let X be a random variable with mean value μ: $E[X] = \mu$, Here the operator E denotes the average or expected value of X. The standard deviation of X is the quantity $\delta = \sqrt{E[(X-\mu)^2]}$.

Throughout our paper, by convention, vectors are assumed to be in column form and are written using bold lower-case letters, e.g. $\mathbf{x}$. For row vectors we use the transpose $\mathbf{x}^T$. We use bold upper-case letters like $\mathbf{R}$ to denote matrices, and sometimes identify a matrix with its ordered set of column vectors. We denote the horizontal concatenation of vectors and/or matrices using a vertical bar, e.g., $[\mathbf{R}|\mathbf{R}\mathbf{x}]$. We sometimes apply functions entry-wise to vectors, e.g., $\lfloor \mathbf{x} \rceil$ rounds each entry of $\mathbf{x}$ to its nearest integer.

Inner products and norms: We denote the standard Euclidean inner product of two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ by $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle = \mathbf{v}_1^T \mathbf{v}_2$. We will be using norms in many of the

inequalities in this work. For $\mathbf{v} = (v_1, \cdots, v_n)^T$ and $p \geq 1$, the l_p norm is $\|\mathbf{v}\|_p = \sqrt[p]{\sum_{i=1}^{n} |v_i|^p}$, the l_∞ norm is $\|\mathbf{v}\|_\infty = \max\{|v_1|, \cdots, |v_n|\}$, the l_1 norm is $\|\mathbf{v}\|_1 = \sum_{i=1}^{n} |v_i|$ and the Euclidean norm is $\|\mathbf{v}\|_2 = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle} = \sqrt{\sum_{i=1}^{n} |v_i|^2}$. For a vector $\mathbf{v}$ we let $\|\mathbf{v}\|$ denote its l_2 norm.

We use the following variant of the leftover hash lemma [19].

Theorem 1. *(Matrix-vector leftover hash lemma [11] Lemma 2.1) Let $\lambda \in \mathbb{N}$, $n \in \mathbb{N}$, $q \in \mathbb{N}$, and $m \geq n \log(q) + 2\lambda$. Let $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{m \times n}$ be a uniformly random matrix, let $\mathbf{r} \xleftarrow{R} \{0,1\}^m$ and $\mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n$, Then:*

$$\Delta\big((\mathbf{A}, \mathbf{A}^T \cdot \mathbf{r}), (\mathbf{A}, \mathbf{y})\big) \leq 2^{-\lambda} \tag{1}$$

where $\Delta(\mathbf{A}, \mathbf{B})$ denotes the statistical distance between the distributions $\mathbf{A}$ and $\mathbf{B}$.

2.2 Discrete Gaussians

The modern approach [28,26] to lattice cryptography relies on Gaussian-like probability distributions over lattices. In our constructions we need to analyze the behavior of error elements sampled from Gaussian distributions. Here we recall the relevant definitions.

Definition 1. *([1] Def.7) Let L be a subset of $\mathbb{Z}^m$. For a vector $\mathbf{c} \in \mathbb{R}^m$ and a positive parameter $\sigma \in \mathbb{R}$, define:*

$$\rho_{\sigma,c}(\mathbf{x}) = \exp\left(-\pi \frac{\|\mathbf{x} - \mathbf{c}\|^2}{\sigma^2}\right) \quad and \quad \rho_{\sigma,c}(L) = \sum_{x \in L} \rho_{\sigma,\mathbf{c}}(x)$$

The discrete Gaussian distribution over L with center c and parameter σ is

$$\forall y \in L,\ D_{L,\sigma,\mathbf{c}}(y) = \frac{\rho_{\sigma,\mathbf{c}}(y)}{\rho_{\sigma,\mathbf{c}}(L)}$$

For notational convenience, $\rho_{\sigma,0}$ and $D_{L,\sigma,0}$ are abbreviated as ρ_σ and D_σ. We write D_σ^m for $D_{\mathbb{Z}^m,\sigma,0}$.

Definition 2. *([11] Def.2.1) (B-bounded distributions). A distribution ensemble $\{\chi_n\}_{n \in \mathbb{N}}$, supported over the integers, is called B-bounded if:*

$$\Pr_{x \xleftarrow{\$} \chi_n}[|x| \geq B] \leq 2^{-\tilde{\Omega}(n)}$$

For a distribution ensemble $\chi = \chi(\lambda)$ over the integers, and integers bounded $B = B(\lambda)$, we say that χ is B-bounded if $Pr_{x \leftarrow \chi(\lambda)}[|x| \leq B(\lambda)]$.

For the analysis of our scheme we require some bounds on the norms of vectors sampled from Gaussian distributions.

Lemma 1. *([21] Lemma 4.4)*

1. *For $\forall\, k > 0$, $Pr[|e| > k\sigma, e \leftarrow D_\sigma^1] \leq 2\exp(-\frac{k^2}{2})$;*
2. *For $\forall\, k > 0$, $Pr[\|\mathbf{e}\| > k\sigma\sqrt{m}, \mathbf{e} \leftarrow D_\sigma^m] \leq k^m \exp(\frac{m}{2}(1 - k^2))$.*

Remark 1. Throughout the paper, we suppose $\sigma \geq 2\sqrt{n}$. Therefore, if $\mathbf{e} \leftarrow D_\sigma^m$ then we have, on average, that $\|\mathbf{e}\| \approx \sqrt{m}\sigma$. Lemma 1 (2) implies that $\|\mathbf{e}\| \leq 2\sigma\sqrt{m}$ with overwhelming probability.

2.3 Learning with Errors

The learning with errors problem is the main computational assumption underlying the GSW cryptosystem and our variant of it.

Definition 3. *(Learning with Errors Distribution) For a vector* $\mathbf{s} \in \mathbb{Z}_q^n$ *called the secret, the LWE distribution* $\mathcal{A}_{\mathbf{s},\chi}$ *over* $\mathbb{Z}_q^n \times \mathbb{Z}_q$ *is sampled by choosing* $\mathbf{a} \in \mathbb{Z}_q^n$ *uniformly at random, choosing* $e \leftarrow \chi$*, and outputting* $\big(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \ (mod\ q)\big)$.

There are two main versions of the LWE problem: search version, which is to find the secret given LWE samples, and decision version, which is to distinguish between LWE samples and uniformly random ones.

Definition 4. *(Search-LW* $E_{n,q,\chi,m}$*) Given* m *independent samples* $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ *drawn from* $\mathcal{A}_{\mathbf{s},\chi}$ *for a uniformly random* $\mathbf{s} \in \mathbb{Z}_q^n$ *(fixed for all samples), find* $\mathbf{s}$*.*

Definition 5. *(Decision-LW* $E_{n,q,\chi,m}$*) Given* m *independent samples* $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ *where every sample is distributed according to either: (1)* $\mathcal{A}_{\mathbf{s},\chi}$ *for a uniformly random* $\mathbf{s} \in \mathbb{Z}_q^n$ *(fixed for all samples), or (2) the uniform distribution, distinguish which is the case (with non-negligible advantage).*

Regev and others [28,26,25,27] show that the LWE problem is as hard as approximating the shortest vector problem in lattices (for appropriate parameters).

The following theorem is a key result used to show the security of our scheme.

Theorem 2. *Let* $m > n \in \mathbb{N}$*, let* $q \in \mathbb{N}$ *and let* χ *be a discrete Gaussian distribution on* $\mathbb{Z}$ *such that the* (n, q, χ, m)*-LWE problem is hard. Let* t *be an integer such that* $t = O(\log(n))$*. Define two distributions* $\mathcal{X}$ *and* $\mathcal{Y}$ *as follows.*

- $\mathcal{X}$ *is the distribution on* $m \times (t + n)$ *matrices*

$$[\mathbf{b}_1| \cdots |\mathbf{b}_t|\mathbf{B}]$$

  where $\mathbf{B} \in \mathbb{Z}_q^{m \times n}$ *is chosen uniformly at random and where, for all* $1 \leq i \leq t$*,*

$$\mathbf{b}_i = \mathbf{B}\mathbf{t}_i + \mathbf{e}_i \pmod q$$

  where $\mathbf{t}_i$ *is sampled uniformly from* $\mathbb{Z}_q^n$ *and* $\mathbf{e}_i$ *is sampled from a discrete Gaussian distribution* χ*.*
- $\mathcal{Y}$ *is the uniform distribution on* $\mathbb{Z}_q^{m \times (t+n)}$*.*

Then the two distributions $\mathcal{X}$ *and* $\mathcal{Y}$ *are computationally indistinguishable.*

Proof. Let D be probabilistic polynomial-time adversary that can distinguish $\mathcal{X}$ from $\mathcal{Y}$ with non-negligible advantage. For $1 \leq i \leq t + 1$ we introduce intermediate distributions $\mathcal{X}_i$ given by

$$[\mathbf{b}'_1| \cdots |\mathbf{b}'_{i-1}|\mathbf{b}_i| \cdots |\mathbf{b}_t|\mathbf{B}]$$

where $\mathbf{b}_i$ is as above and $\mathbf{b}'_i$ is uniformly chosen from $\mathbb{Z}_q^m$. Hence $\mathcal{X}_1 = \mathcal{X}$ and $\mathcal{X}_{t+1} = \mathcal{Y}$.

By assumption, D can distinguish $\mathcal{X}_1$ from $\mathcal{X}_{t+1}$ with noticeable advantage ϵ and so, by a standard hybrid argument, there is some i such that D can distinguish $\mathcal{X}_i$ from $\mathcal{X}_{i+1}$ with some noticeable advantage at least ϵ/t.

It is then easy to see that D gives an LWE distinguisher: given an LWE challenge $(\mathbf{B}, \mathbf{y})$ one samples $\mathbf{b}'_1, \ldots, \mathbf{b}'_{i-1}$ uniformly and samples $\mathbf{b}_{i+1}, \ldots, \mathbf{b}_t$ as specified above (so they are from an LWE distribution, all for different choices of the secret vector) and then calls D on

$$[\mathbf{b}'_1|\cdots|\mathbf{b}'_{i-1}|\mathbf{y}|\mathbf{b}_{i+1}|\cdots|\mathbf{b}_t|\mathbf{B}].$$

By assumption, no such distinguisher exists. Hence the theorem is proved. □

2.4 Leveled homomorphic Encryption

Definition 6. *Fix a function $L = L(\lambda)$. An L-leveled homomorphic encryption scheme for a class of circuits $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbf{N}}$ of depth L consists of four polynomial-time algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *such that:*

- *The key-generation algorithm* KeyGen *is a randomized algorithm that takes the security parameter 1^λ as input and outputs a public key* pk *and secret key* sk.
- *The encryption algorithm* Enc *is a randomized algorithm that takes a public key* pk *and a message $m \in \{0,1\}$ as input, and outputs a ciphertext c.*
- *The decryption algorithm* Dec *is a deterministic algorithm that takes the secret key* sk *and a ciphertext c as input, and outputs a message $m \in \{0,1\}$.*
- *The homomorphic evaluation algorithm* Eval *takes as input a public key* pk*, a circuit $C \in \mathcal{C}_\lambda$, and a list of ciphertexts $c_1, \cdots, c_{l(\lambda)}$, it outputs a ciphertext c^*.*

The following correctness properties are required to hold:

- For any λ, any $m \in \{0,1\}$, and any $(\mathsf{pk}, \mathsf{sk})$ output by $\mathsf{KeyGen}(1^\lambda)$, we have
$$m = \mathsf{Dec}(\mathsf{sk}, (\mathsf{Enc}(\mathsf{pk}, m))).$$
- For any λ, any $m_1, \cdots, m_l$, and any $C \in \mathcal{C}_\lambda$, we have
$$C(m_1, \cdots, m_l) = \mathsf{Dec}(\mathsf{sk}, (\mathsf{Eval}(\mathsf{pk}, C, \mathsf{Enc}(\mathsf{pk}, m_1), \cdots, \mathsf{Enc}(\mathsf{pk}, m_l)))).$$

We use the standard notion of security against chosen-plaintext attacks.

Definition 7. *A homomorphic encryption scheme is secure against chosen-plaintext attacks (also called IND-CPA-secure) if for any polynomial-time adversary $\mathcal{A}$ the following is negligible in λ:*

$$|Pr[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, 0)) = 1] - Pr[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, 1)) = 1]|,$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

The notion of CCA1 security considers that the adversary $\mathcal{A}$ is given a public key and then has access to a decryption oracle in the first phase of its attack. It can ask for decryptions of any inputs of its choosing. In the second phase the adversary is given a challenge ciphertext $\mathsf{Enc}(\mathsf{pk}, b)$ and can no longer make queries to the decryption oracle. The CCA1 security model is intended for analysing if an adversary can learn the private key from making decryption queries. It is an appropriate model for studying homomorphic encryption.

2.5 Basic Tools

We now recall some basic tools that were introduced by Brakerski, Gentry and Vaikuntanathan [9] and used in many more works such as [10,8,18].

Fix $q, m \in \mathbb{N}$. Let $l = \lfloor \log(q) \rfloor + 1$, so that $2^{l-1} \leq q < 2^l$, and $N = m \cdot l$.

Definition 8. *The algorithm* $\mathsf{PowersOf2}$ *takes an m-dimensional vector $\mathbf{v} \in \mathbb{Z}_q^m$ and outputs an N-dimensional vector*

$$\left(v_1, 2v_1, \cdots, 2^{l-1}v_1, \cdots, v_m, 2v_m, \cdots, 2^{l-1}v_m\right)^T$$

in $\mathbb{Z}_q^N$.

Definition 9. *The algorithm* $\mathsf{BitDecomp}$ *takes as input a vector $\mathbf{v} \in \mathbb{Z}_q^m$ and outputs an N-dimensional vector $(v_{1,0}, \cdots, v_{1,l-1}, \cdots, v_{m,0}, \cdots, v_{m,l-1})^T \in \{0,1\}^N$ where $v_{i,j}$ is the j-th bit in v_i's binary representation (ordered from least significant to most significant.) In other words,*

$$v_i = \sum_{j=0}^{l-1} 2^j v_{i,j}.$$

Definition 10. *The algorithm* $\mathsf{BitDecomp}^{-1}$ *takes as input a vector*

$$\mathbf{v} = (v_{1,0}, \cdots, v_{1,l-1}, \cdots, v_{m,0}, \cdots, v_{m,l-1})^T \in \mathbb{Z}_q^N$$

and outputs the vector $(\sum_{j=0}^{l-1} 2^j \cdot v_{1,j}, \cdots, \sum_{j=0}^{l-1} 2^j \cdot v_{m,j})^T \in \mathbb{Z}_q^m$. Note that the input vectors $\mathbf{v}$ need not be binary, the algorithm is well-defined for any input vector in $\mathbb{Z}^N$.

Definition 11. *The algorithm* $\mathsf{Flatten}$ *takes as input a vector $\mathbf{v} \in \mathbb{Z}_q^N$ and outputs an N-dimensional binary vector (i.e. an element of $\{0,1\}^N$). It is defined by* $\mathsf{Flatten}(\mathbf{v}) = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\mathbf{v}))$.

The following straightforward facts are given in [18].

Proposition 1. *Let $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_q^m$ and let $\mathbf{a}' \in \mathbb{Z}_q^N$. Then $\langle \mathsf{BitDecomp}(\mathbf{a}), \mathsf{PowersOf2}(\mathbf{b}) \rangle = \langle \mathbf{a}, \mathbf{b} \rangle$ and*

$$\begin{aligned}
\langle \mathbf{a}', \mathsf{PowersOf2}(\mathbf{b}) \rangle &= \langle \mathsf{BitDecomp}^{-1}(\mathbf{a}'), \mathbf{b} \rangle \\
&= \langle \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\mathbf{a}')), \mathsf{PowersOf2}(\mathbf{b}) \rangle \\
&= \langle \mathsf{Flatten}(\mathbf{a}'), \mathsf{PowersOf2}(\mathbf{b}) \rangle.
\end{aligned}$$

These algorithms can be extended from vectors to matrices in the natural way.

One can express the above functions in terms of the gadget matrix $\mathbf{G}$ from Micciancio and Peikert [23]. Define $\mathbf{G} = \mathbf{I}_m \otimes \mathbf{g} \in \mathbb{Z}_q^{m \times N}$ where $\mathbf{g} = (1, 2, 4, \ldots, 2^{l-1})^T$. For $\mathbf{v} \in \mathbb{Z}_q^m$ we have $\mathsf{PowersOf2}(\mathbf{v}) = \mathbf{v}^T\mathbf{G}$. For $\mathbf{v} \in \mathbb{Z}_q^N$ we have $\mathsf{BitDecomp}^{-1}(\mathbf{v}) = \mathbf{G}\mathbf{v}$. For $\mathbf{a} \in \mathbb{Z}_q^m$ the algorithm $\mathsf{BitDecomp}(\mathbf{a})$ can be renamed as $\mathbf{G}^{-1}(\mathbf{a})$. The above definitions and results can therefore be expressed in the language of $\mathbf{G}$ and $\mathbf{G}^{-1}$ as follows.

Lemma 2. *([23] and [25] Lemma 2.1) For any $N \geq m\lceil \log q \rceil$ there exists a fixed efficiently computable matrix $\mathbf{G} \in \mathbb{Z}_q^{m \times N}$ and an efficiently computable deterministic "short preimage" function $\mathbf{G}^{-1}(\cdot)$ satisfying the following. On input a matrix $\mathbf{M} \in \mathbb{Z}_q^{m \times m'}$ for any m'. the inverse function $\mathbf{G}^{-1}(\mathbf{M})$ outputs a matrix $\mathbf{G}^{-1}(\mathbf{M}) \in \{0,1\}^{N \times m'}$ such that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{M}) = \mathbf{M}$.*

We can think of $\mathbf{G}$ as a special matrix with a "public trapdoor" that allows us to solve the *short integer solution* SIS problem. We stress that $\mathbf{G}^{-1}(\cdot)$ is not itself a matrix but rather an efficiently computable function.

3 Gentry-Sahai-Waters Homomorphic Encryption

In this section we describe the Gentry-Sahai-Waters (GSW13) homomorphic encryption scheme, then we sketch an adaptive attack on it similar to one due to Chenal and Tang [12].

3.1 GSW13 Scheme

Let k be a security parameter and let L be the number of levels for the somewhat homomorphic scheme. We describe the algorithms that form the GSW13 scheme [18]. The algorithm is originally defined in terms of the functions BitDecomp, $\mathsf{BitDecomp}^{-1}$ and Flatten, but we tend to follow the formulation in [4,25] and so use the matrix $\mathbf{G}$.

- $\mathsf{GSW.Setup}(1^k, 1^L)$:
  1. Choose a modulus q of $\kappa = \kappa(k, L)$ bits, parameter $n = n(k, L) \in \mathbb{N}$, and error distribution $\chi = \chi(k, L)$ on $\mathbb{Z}$ so that the (q, n, χ)-LWE problem achieves at least 2^k security against known attacks.
     Choose a parameter $m = m(k, L) = O(n \log(q))$;
  2. Output: $\mathsf{params} = (n, q, \chi, m)$.
     We also use the notation $l = \lfloor \log(q) \rfloor + 1$ and $N = (n + 1) \cdot l$.
- $\mathsf{GSW.KeyGen}(\mathsf{params})$:
  1. Sample uniformly $\mathbf{t} = (t_1, \ldots, t_n)^T \leftarrow \mathbb{Z}_q^n$ and compute
  $$\mathbf{s} \leftarrow (1, -\mathbf{t}^T)^T = (1, -t_1, \cdots, -t_n)^T \in \mathbb{Z}_q^{(n+1)\times 1};$$
  2. Generate a matrix $\mathbf{B} \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly and a vector $\mathbf{e} \leftarrow \chi^m$;
  3. Compute $\mathbf{b} = \mathbf{B}\mathbf{t} + \mathbf{e} \in \mathbb{Z}_q^m$ and construct the matrix $\mathbf{A} = (\mathbf{b} \mid \mathbf{B}) \in \mathbb{Z}_q^{m \times (n+1)}$ as the vector $\mathbf{b}$ followed by the n columns of $\mathbf{B}$.
     Observe that
     $$\mathbf{A}\mathbf{s} = (\mathbf{b} \mid \mathbf{B})\mathbf{s} = (\mathbf{B}\mathbf{t} + \mathbf{e} \mid \mathbf{B}) \begin{pmatrix} 1 \\ -\mathbf{t} \end{pmatrix} = \mathbf{B}\mathbf{t} + \mathbf{e} - \mathbf{B}\mathbf{t} = \mathbf{e}.$$
  4. Return $\mathsf{sk} \leftarrow \mathbf{s}$ and $\mathsf{pk} \leftarrow \mathbf{A}$.
- $\mathbf{C} \leftarrow \mathsf{GSW.Enc}(\mathsf{params}, \mathsf{pk}, \mu)$: In order to encrypt one-bit messages $\mu \in \{0, 1\}$:
  1. Let $\mathbf{G}$ be the $(n + 1) \times N$ gadget matrix as above;

2. Sample uniformly a matrix $\mathbf{R} \leftarrow \{0,1\}^{m \times N}$;
3. Compute $\mathbf{C} = \mu\mathbf{G} + \mathbf{A}^T\mathbf{R} \pmod q \in \mathbb{Z}_q^{(n+1)\times N}$;
   In the original GSW paper this was written as $\mathsf{Flatten}(\mu\mathbf{I}+\mathsf{BitDecomp}(\mathbf{RA})) \in \{0,1\}^{N\times N}$ where $\mathbf{I}$ is an identity matrix.

- $\mu' \leftarrow \mathsf{GSW.Dec}(\mathsf{params}, \mathsf{sk}, \mathbf{C})$:
  1. We have $\mathsf{sk} = \mathbf{s} \in \mathbb{Z}_q^{n+1}$;
  2. Let I be such that $q/4 < 2^{I-1} \leq q/2$. Let $\mathbf{C}_I$ be the I-th column of $\mathbf{C}$;
  3. Compute $x \leftarrow \langle \mathbf{C}_I, \mathbf{s}\rangle \pmod q$ in the range $(-q/2, q/2]$;
     Note that $\langle \mathbf{C}_I, \mathbf{s}\rangle = \mathbf{C}_I^T\mathbf{s}$ and that
     $$\begin{aligned}\mathbf{C}^T\mathbf{s} &= \mu\mathbf{G}^T\mathbf{s} \ + \ \mathbf{R}^T\mathbf{A}\mathbf{s} \\ &= \mu(1,2,4,\cdots)^T \ + \ \mathbf{R}^T\mathbf{e}\end{aligned}$$
     and selecting the I-th column of $\mathbf{C}$ corresponds to selecting the I-th coordinate of this vector, which is $\mu 2^{I-1} + \mathbf{R}_I^T\mathbf{e}$.
  4. Output $\mu' = ||\lfloor x/2^{I-1}\rceil||$.
     So if $|x| < 2^{I-2} \leq q/4$ then return 0 and if $|x| > 2^{I-2}$ then return 1.

- $\mathsf{GSW.Eval}(\mathsf{params}, \mathbf{C}_1, \cdots, \mathbf{C}_l)$:
  - $\mathsf{GSW.Add}(\mathbf{C}_1, \mathbf{C}_2)$: output
    $$\mathbf{C}_1 + \mathbf{C}_2 = (\mu_1+\mu_2)\mathbf{G} + \mathbf{A}^T(\mathbf{R}_1+\mathbf{R}_2) \in \mathbb{Z}_q^{(n+1)\times N};$$
  - $\mathsf{GSW.Mult}(\mathbf{C}_1, \mathbf{C}_2)$: Compute $\mathbf{G}^{-1}(\mathbf{C}_2) \in \{0,1\}^{N\times N}$ and output $\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$.
    Note that
    $$\begin{aligned}\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2) &= \big(\mu_1\mathbf{G} + \mathbf{A}^T\mathbf{R}_1\big)\mathbf{G}^{-1}(\mathbf{C}_2) \\ &= \mu_1\mathbf{C}_2 + \mathbf{A}^T\mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) \\ &= \mu_1\mu_2\mathbf{G} + \mathbf{A}^T\mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1\mathbf{A}^T\mathbf{R}_2 \\ &= \mu_1\mu_2\mathbf{G} + \mathbf{A}^T\big(\mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1\mathbf{R}_2\big) \in \mathbb{Z}_q^{(n+1)\times N}.\end{aligned}$$

  One may also compute a homomorphic NAND gate by outputting $\mathbf{G} - \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$.

Remark 2. Note that the formulation of the decryption algorithm in [25] is to choose an appropriate vector $\mathbf{w}$ and compute $\mathbf{s}\mathbf{C}\mathbf{G}^{-1}(\mathbf{w}^T)$. This is considerably less efficient than the original GSW decryption algorithm (both in terms of computation time and also the size of the error term). Hence we employ the original GSW decryption algorithm for our scheme.

There is also a variant of the scheme that handles messages in $\mathbb{Z}_q$ when q is a power of two. We refer to [18] for the details.

3.2 Security

A sketch proof is given in [18] of the following theorem.

Theorem 3. *Let (n, q, χ) be such that the $\mathsf{LWE}_{(n,q,\chi)}$ assumption holds and let $m = O(n\log(q))$. Then the GSW13 scheme is IND-CPA secure.*

The main step in the proof is showing that $(\mathbf{A}, \mathbf{RA})$ is computationally indistinguishable from uniform.

3.3 Key Recovery Attacks

We now explain two adaptive attacks on the GSW scheme that allow to determine the private key. Attacks of this type are well-known and are outside the security model of the original paper [18].

Adaptive Attack 1: We now briefly review an adaptive key recovery attack similar to the one by Chenal and Tang [12]. The adversary recovers the secret key $\mathbf{s} = (1, -\mathbf{t}^T)^T$ through a number of decryption oracle queries.

The attacker will call the decryption oracle on a matrix $\mathbf{C}$ of their choice, and the oracle will return the "most significant bit" of $\langle \mathbf{C}_I, \mathbf{s} \rangle$ where $\mathbf{C}_I$ is the I-th column of $\mathbf{C}$ and I is known to the adversary. Technically it computes $\lfloor \mathbf{C}_I^T \mathbf{s} \pmod q / 2^{I-1} \rceil$, and this is essentially the most significant bit in the representation (ignoring signs).

The attack is therefore quite simple: one chooses $\mathbf{C}_I = (0, 0, \ldots, 0, M, 0, \ldots, 0)^T$ for appropriate values M in appropriate positions and learns the entries of the secret key bit-by-bit. For example, to compute $t_1 \in \mathbb{Z}_q$ one makes a decryption oracle query on a matrix with $\mathbf{C}_I = (0, 1, 0, \ldots, 0)^T$. Hence

$$\langle \mathbf{C}_I, \mathbf{s} \rangle = -t_1$$

and so one learns the most significant bit of t_1. One can now re-scale (e.g., choose $\mathbf{C}_I = (0, 2, 0, \ldots, 0)$) to learn the most significant bit of $2(-t_1) \pmod q$, which yields information about the next most significant bit (one has to take into account the modular reduction if the most significant bit is 1). To separate positive and negative values one can use vectors like $\mathbf{C}_I = (M, 1, 0, \ldots, 0)$, which provides the most significant bit of $\langle \mathbf{C}_I, \mathbf{s} \rangle = M - t_1$. We omit further details here, and refer to [12,13,15] for discussion of these sorts of attacks.

In the above description we have spoken of calling the decryption oracle on matrices whose I-th column is a unit vector. Hence one might think that the attacks can be avoided by rejecting any ciphertext of this form. But since the scheme is homomorphic, one can always add a random encryption of zero to the ciphertext, so that the matrix $\mathbf{C}$ looks just like any other matrix that might be passed to the decryption algorithm. One can consider other variants of the decryption algorithm that try to determine whether a ciphertext is "correctly formed", as was done by Loftus et al [20], but instead we consider a different approach to prevent such attacks.

Adaptive Attack 2: Attack 1 aimed to learn the value $\mathbf{t}$ in $\mathbf{b} = \mathbf{B}\mathbf{t} + \mathbf{e}$. However, if one can compute $\mathbf{e}$ then, using $\mathbf{B}$ and $\mathbf{b}$ then one can also determine $\mathbf{t}$. So in this section we give a method to determine $\mathbf{e}$. The ideas in this section were pointed out to us by Damien Stehlé.

To learn e_j for any $1 \leq j \leq m$ we consider the j-th row of $\mathbf{A}$, which we write as $\mathbf{a}_j = (\mathbf{b}_j \mathbf{t} + e_j \pmod q \mid \mathbf{b}_j) \in \mathbb{Z}_q^{1 \times (n+1)}$. We need to insert this row into the I-th column of the ciphertext matrix, so set $\mathbf{C}_I = \mathbf{a}_j^T$ and set the remaining columns of $\mathbf{C}$ to be zero (again, this presentation is for simplicity, one can add to this matrix a random encryption of zero to "hide" the attack).

Now, the decryption oracle computes

$$\begin{aligned}\langle \mathbf{C}_I, \mathbf{s} \rangle &= \mathbf{C}_I^T \mathbf{s} = \mathbf{a}_j \mathbf{s} \\ &= (\mathbf{b}_j \mathbf{t} + e_j \pmod q \mid \mathbf{b}_j), (1, -\mathbf{t}^T)^T \rangle \\ &= \mathbf{b}_j \mathbf{t} + e_j - \mathbf{b}_j \mathbf{t} = e_j\end{aligned}$$

and so returns $\lfloor \lfloor e_j / 2^{I-1} \rceil \rfloor$. One can therefore learn the most significant bit of e_j (which is certainly 0 since e_j is supposed to be small relative to q).

To extend the attack, choose an integer $-q/2 < u < q/2$ and repeat the attack on $\mathbf{a}'_j = \mathbf{a}_j + (u \mid 0, \cdots, 0)$. The decryption oracle therefore returns $\lfloor \lfloor (e_j + u \pmod q)/2^{I-1} \rceil \rfloor$ and we can learn whether $|e_j + u| < 2^{I-2}$ or not.

By trying $u = 1, 2, 4, \cdots, 2^k$ one can determine the smallest power of two such that $e_j + 2^k \geq 2^{I-2}$. This implies that $2^{I-2} - 2^k \leq e_j < 2^{I-2} - 2^{k-1}$ and a binary search style algorithm allows to determine e_j exactly using about k further decryption oracle queries.

4 Multiple Secret Scheme (MGSW)

We now describe our variant of the GSW13 scheme. First we give some motivation for our design. Adaptive attack 1 exploits the fact that certain queries to the decryption oracle leak one bit of one component of the fixed secret key $\mathbf{s} = (1, -t_1, \ldots, -t_n)^T$. Our main idea is to have a large set of possible secret keys. Each execution of the decryption algorithm will generate a fresh random "one-time" secret key $\mathbf{s}$. The decryption algorithm itself does not change, and we do not introduce any notion of "valid ciphertext", so an attacker can still learn one bit of one component of the key used for decryption. However, the main idea of our approach is that an attacker cannot iterate adaptive attack 1 to learn an "entire" secret key, since each query gives information about a fresh random key and these keys are uncorrelated with each other.

To achieve this we need to modify the key generation. Instead of choosing $\mathbf{A}$ of the form $[\mathbf{B}\mathbf{t} + \mathbf{e} | \mathbf{B}]$ for a uniform $\mathbf{t}$ and a short vector $\mathbf{e}$, we construct

$$\mathbf{A}' = [\mathbf{B}\mathbf{t}_1 + \mathbf{e}_1 | \mathbf{B}\mathbf{t}_2 + \mathbf{e}_2 | \cdots | \mathbf{B}\mathbf{t}_t + \mathbf{e}_t | \mathbf{B}]$$

where $\mathbf{t}_1, \ldots, \mathbf{t}_t$ are sampled uniformly in $\mathbb{Z}_q^n$ and $\mathbf{e}_1, \ldots, \mathbf{e}_t$ are sampled from the discrete Gaussian distribution χ^m. We will use the notation $\mathbf{e}_i = (e_{i,1}, \ldots, e_{i,m})^T$. We now define secret keys

$$\mathbf{s}_1 = (1, 0, \ldots, 0, -\mathbf{t}_1^T)^T, \ldots, \mathbf{s}_t = (0, \ldots, 0, 1, -\mathbf{t}_t^T)^T$$

so that $\mathbf{A}'\mathbf{s}_i = \mathbf{e}_i \pmod q$ for all $1 \leq i \leq t$. Finally, every time the decryption algorithm runs, it generates a fresh random one-time secret

$$\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{s}_i.$$

The point is that

$$\mathbf{e}' = \mathbf{A}'\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{A}'\mathbf{s}_i = \sum_{i=1}^{t} \lambda_i \mathbf{e}_i$$

is a short vector if the integers λ_i are small (e.g., one could take the $\lambda_i \in \{0, 1\}$ or $\{-1, 0, 1\}$ or from a discrete Gaussian distribution. There are at least 2^t possible secret keys, so t does not have to be very large to ensure that no secret key is ever used more than once in a practical scheme.

We now give the formal details.

4.1 MGSW Scheme

- $\mathsf{params} \leftarrow \mathsf{MGSW.Setup}(1^k, 1^L)$
  1. Identical to GSW.Setup algorithm except that a parameter $t = O(\log(n))$ is chosen (the number of secret keys);
  2. Output $\mathsf{params} = (n, q, \chi, m, t)$ and let $l = \lfloor \log(q) \rfloor + 1$ and $N = (t + n) \cdot l$.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{MGSW.KeyGen}(\mathsf{params})$:
  1. Sample uniformly $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n, i \in [t]$ and output

     $$\mathbf{s}_i \leftarrow (\mathbf{I}_i \mid -\mathbf{t}_i^T)^T = (0, \cdots, 1, \cdots, 0, -t_{i,1}, \cdots, -t_{i,n})^T \in \mathbb{Z}_q^{n+t},$$

     where $\mathbf{I}_i$ is the i-th row of the $t \times t$ identity matrix $\mathbf{I}$ and so the i-th coordinate of $\mathbf{s}_i$ equals 1;
  2. Choose a matrix $\mathbf{B} \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly and t vectors $\mathbf{e}_i \leftarrow \chi^m$ for $i \in [t]$;
  3. Compute $\mathbf{b}_i = \mathbf{B}\mathbf{t}_i + \mathbf{e}_i \in \mathbb{Z}_q^m$ and $\mathbf{A}' = [\mathbf{b}_1 | \cdots | \mathbf{b}_t | \mathbf{B}] \in \mathbb{Z}_q^{m \times (n+t)}$;
  4. Output $\mathsf{pk} \leftarrow \mathbf{A}'$ and $\mathsf{sk} \leftarrow \{\mathbf{s}_1, \ldots, \mathbf{s}_t\}$.
- $\mathbf{C} \leftarrow \mathsf{MGSW.Enc}(\mathsf{params}, \mathsf{pk}, \mu)$:
  1. To encrypt a message $\mu \in \mathbb{Z}_q$, sample uniformly a matrix $\mathbf{R}' \in \{0, 1\}^{m \times N}$;
  2. Compute and output the ciphertext $\mathbf{C} = \mu\mathbf{G} + \mathbf{A}'^T\mathbf{R}' \in \mathbb{Z}_q^{(n+t) \times N}$, where the $\mathbf{G}$ denotes the $(n + t) \times N$-dimensional gadget matrix.
- $\mu' \leftarrow \mathsf{MGSW.Dec}(\mathsf{params}, \mathsf{sk}, \mathbf{C})$:
  1. Choose $\lambda_1, \ldots, \lambda_t$ uniformly from $\{0, 1\}$ such that they are not all zero, and generate a one-time key $\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{s}_i$;
  2. Choose an integer $1 \leq i \leq t$ such that $\lambda_i = 1$ and set $I = (i - 1)l + j$ such that the i-th entry of the I-th column of G is 2^{j-1} where $q/4 < 2^{j-1} \leq q/2$;
  3. Compute $x = \langle \mathbf{C}_I, \mathbf{s} \rangle \pmod q = \mathbf{C}_I^T \mathbf{s} \pmod q$ in the range $(-q/2, q/2]$;
  4. Output $\mu' = |\lfloor x/2^{j-1} \rceil|$.
- The homomorphic operations are exactly the same as in the original scheme.

4.2 Correctness and Homomorphic Operations

In this section, we will analyze the scheme's correctness and homomorphic operations, following the arguments from [18].

The main change in the scheme is that a secret key is changed from $\mathbf{s} = (1, -\mathbf{t}^T)^T$ to a large set of secret keys of the form $\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{s}_i$ with $\lambda_i \in \{0, 1\}$. We have $\mathbf{A}'\mathbf{s}_i = \mathbf{e}_i \pmod q$ for all $1 \leq i \leq t$ where $\mathbf{e}_i$ is chosen from a discrete Gaussian distribution. Writing $\mathbf{e}' = \mathbf{A}'\mathbf{s}' \pmod q$ for any choice of one-time secret key $\mathbf{s}'$ we have

$$\|\mathbf{e}'\| = \|\mathbf{A}'\mathbf{s}'\| = \left\| \sum_{i=1}^{t} \lambda_i \mathbf{e}_i \right\| \leq \sum_{i=1}^{t} |\lambda_i| \|\mathbf{e}_i\|.$$

Since we may assume $\|\mathbf{e}_i\| \leq 2\sqrt{m}\sigma$ it follows that $\|\mathbf{e}'\| \leq 2t\sqrt{m}\sigma$. Gentry, Sahai and Waters consider B-bounded error vectors to show that decryption is correct. If $\|\mathbf{e}_i\|_\infty = \max\{|e_{i,j}|\} \leq B$ then, by the same argument $\|\mathbf{e}'\|_\infty \leq B' = tB$.

To specify parameters we first fix a level L. The parameter n determines a number of parameters $\sigma \geq 2\sqrt{n}$, $t = O(\log(n))$, $B = 10\sigma$, $B' = tB$. It is necessary to choose (l, q) with $2^{l-1} < q < 2^l$ and $q > 8B'((t+n)l + 1)^{L+1}$. Finally, one selects a large enough parameter n so that the $(n, q, D_\sigma)-\mathsf{LWE}$ problem is hard with these choices for q and σ. Note that $m > 2n\log(q) > t + n$ and $N = (t + n)l$. These are the parameters used to describe the MGSW key generation.

We say that a ciphertext $\mathbf{C}$ is at *level* i if it has been formed by running the Evaluate algorithm at most i times on encryptions of messages. The Encrypt algorithm outputs ciphertexts of level 0.

Lemma 3. *Let notation and parameters be as above. Let* $\mathbf{C}$ *be any ciphertext at level* $i \leq L$*. Then the decryption algorithm returns the correct message* μ*.*

Proof. Let $\mathbf{s}'$ be any one-time key for the MGSW scheme, so that $\mathbf{A}'\mathbf{s}' = \mathbf{e}' \pmod q$ where $\|\mathbf{e}'\|_\infty \leq B'$. Write $\mathbf{v}' = \mathbf{G}^T\mathbf{s}' = \mathsf{PowersOf2}(\mathbf{s}')$. We prove the lemma by induction.

First we consider level 0 ciphertexts $\mathbf{C}$. We have

$$\begin{aligned} \mathbf{C}^T\mathbf{s}' &= \mu\mathbf{G}^T\mathbf{s}' + \mathbf{R}'^T\mathbf{A}\mathbf{s}' \\ &= \mu\mathbf{G}^T\mathbf{s}' + \mathbf{R}'^T\mathbf{e}' \\ &= \mu\mathbf{v}' + \sum_{i}^{t} \lambda_i \mathbf{R}'^T\mathbf{e}_i \pmod q \end{aligned}$$

The infinity norm of $\mathbf{R}'^T\mathbf{e}_i$ is upper bounded by $NB' < q/8 < 2^{j-2}$, hence decryption is correct.

We now give the inductive step. Suppose we have two ciphertexts $\mathbf{C}_1$ and $\mathbf{C}_2$ at level $\leq i$, that encrypt messages μ_1 and $\mu_2 \in \{0, 1\}$, respectively. (Please excuse the abuse of notation.) By the inductive hypothesis, for $j \in \{1, 2\}$ we have

$$\mathbf{C}_j^T\mathbf{s}' = \mu_j\mathbf{v}' + E_j$$

where $E_j = {\mathbf{R}'_j}^T \mathbf{e}' = \sum_i^t \lambda_i {\mathbf{R}'_j}^T \mathbf{e}_i^T$ satisfies $\|E_j\|_\infty \leq t\|\mathbf{e}^T\|_\infty \|\mathbf{R}'\|_\infty \leq (N+1)^{i+1}B'$. We consider the case of addition, and write $\mathbf{C}^{Add} = \mathbf{C}_1 + \mathbf{C}_2$. Then $(\mathbf{C}^{Add})^T\mathbf{s}' = (\mu_1+\mu_2)\mathbf{v}' + (E_1+E_2)$. Since $\|E_1+E_2\|_\infty \leq 2(N+1)^{i+1}B' < (N+1)^{i+2}B'$ the result follows.

We now consider ciphertext multiplication, which results in ciphertext $\mathbf{C}^{Mult} = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$ such that

$$\mathbf{C}^{Mult} = \mu_1\mu_2\mathbf{G} + \mathbf{err},$$

where $\mathbf{err} := \mu_1 {\mathbf{A}'}_2^T \mathbf{R}'_2 + {\mathbf{A}'_1}^T \mathbf{R}'_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)$. So

$$(\mathbf{C}^{Mult})^T\mathbf{s}' = \mu_1\mu_2\mathbf{G}^T\mathbf{s}' + \mu_1\mathbf{R}_2^T\mathbf{e}'_2 + \mathbf{G}^{-1}(\mathbf{C}_2)^T\mathbf{R}_1^T\mathbf{e}'_1 \pmod q,$$

We have that $\mathbf{C}_2$ is an $(n+t)\times N$ binary matrix and $\mu_1 \in \{0,1\}$ and so

$$\|\mu_1 E_1 + \mathbf{G}^{-1}(\mathbf{C}_2)^T E_2\|_\infty \leq \|E_1\|_\infty + N\|E_2\|_\infty \leq (N+1)^{i+2}B'.$$

The case of NAND is similar: Let $\mathbf{C} = \mathbf{G} - \mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)$. Note that

$$\mathbf{C}^T\mathbf{s}' = \mathbf{G}^T\mathbf{s}' - (\mathbf{C}^{Mult})^T\mathbf{s}'.$$

By the same argument as above this has error bounded by $(N+1)^{i+2}B'$.

Hence, after L operations have been performed, the error terms are bounded by $(N+1)^{L+1}B' < q/8$ and so decryption works correctly. □

4.3 IND-CPA Security

We show the scheme is IND-CPA secure based on the LWE assumption by using Theorem 2 to show that the scheme is indistinguishable from the original GSW13 scheme, and then applying Theorem 3.

Theorem 4. *Let $params = (n, q, \chi, m, t)$ be such that the $LWE_{n,q,\chi,m}$ assumption holds and $m = O(n\log(q))$. Then the MGSW scheme is IND-CPA secure.*

Proof. The proof of security consists of two steps:

- Firstly, we apply Theorem 2 to show that, under the LWE assumption, the matrix $\mathbf{A}' = [\mathbf{b}_1, \cdots, \mathbf{b}_t, \mathbf{B}] \in \mathbb{Z}_q^{m\times(n+t)}$ is computationally indistinguishable from a randomly chosen matrix.
- Then we apply the arguments from the proof of Theorem 3, namely that $\mathbf{A}'^T \cdot \mathbf{R}'$ is indistinguishable from uniform assuming the hardness of $LWE_{n,q,\chi,m}$.

This completes the sketch of the proof. □

5 Adaptive attacks on the Multiple Secret Scheme

5.1 Security Against Adaptive Attack 1

In this section we discuss how Adaptive Attack 1 on the GSW scheme is prevented by our countermeasure.

The crux of our argument is that the one-time keys are distributed uniformly from the point of view of the decryption oracle, and so are independent of the actual secret basis. In terms of linear algebra (assuming for the moment that q is prime), the one-time keys all lie in a vector subspace K of dimension t inside the much larger space $\mathbb{Z}_q^n$. (Not all elements of the space K are valid secret keys; only the ones that correspond to short linear combinations of the basis are allowed.) However, the attacker just gets a single bit of an inner product of the one-time key with the vector coming from the ciphertext. One can think of the inner product with $\mathbf{C}_I$ as giving a projection (linear map) $L_{\mathbf{C}_I} : \mathbb{Z}_q^n \to \mathbb{Z}_q$. So the adversary only sees one bit of one projection of the secret. Even though the subspace K is small, the probability that K lies in the kernel of this projection is equal to the probability that $\mathbf{C}_I$ is chosen in the orthogonal complement $K^\perp$ of K. Since K has dimension t in an n-dimensional space, the dimension of $K^\perp$ is $n - t$. Hence the probability that a randomly chosen $\mathbf{C}_I$ is such that K is in the kernel of the projection $L_{\mathbf{C}_I}$ is $q^{n-t}/q^n = 1/q^t$, which will be negligible. Of course, $\mathbf{C}_I$ is not random but is chosen by the adversary. However, the adversary does not know K and so is unlikely to be able to do better than choosing random vectors. In our argument we will therefore assume that the projection is surjective. Under this assumption it suffices to argue that the distribution of the projected value is close to uniform and so is *independent on the secret vectors*. This is sufficient to deduce that the attack cannot work, since the information revealed by the decryption oracle is therefore independent of the choice of secret keys.

First note that the one-time secret key is of the form

$$\sum_{i=1}^{t} \lambda_i \mathbf{s}_i = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_t \\ \hline \sum_i^t \lambda_i \mathbf{t}_i \end{pmatrix} \in \mathbb{Z}_q^n.$$

The first t entries carry no information about the long-term secret $\mathbf{t}_1, \ldots, \mathbf{t}_t$.

We now fix a linear map $L : \mathbb{Z}_q^n \to \mathbb{Z}_q$ (corresponding to an inner product with $\mathbf{C}_I$). We assume that t' of the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_t$ do not lie in $\ker(L)$ where $t' \approx t$ (a random vector lies in $\ker(L)$ with probability $1/q$ so a given set of l vectors lie in $\ker(L)$ with probability $1/q^l$). When making a decryption oracle query with ciphertext $\mathbf{C}_I$ the adversary gets one bit of information about the value

$$L\left(\sum_{i=1}^{t} \lambda_i \mathbf{s}_i\right) = \sum_{i=1}^{t} \lambda_i L(\mathbf{s}_i).$$

Since $\mathbf{t}_1, \ldots, \mathbf{t}_t$ are sampled uniformly from $\mathbb{Z}_q^n$ we can model $L(\mathbf{s}_1), \ldots, L(\mathbf{s}_t)$ as corresponding to t' non-zero values uniformly sampled from $\mathbb{Z}_q$.

We now want to argue that the information leaked from an oracle query cannot provide any information about the secret values $\mathbf{s}_i$. Our argument is inspired by the left-over hash lemma (Theorem 1) in the one-dimensional case. The left-over hash lemma states that the distribution $(L(\mathbf{s}_1), \ldots, L(\mathbf{s}_t), \sum_i \lambda_i L(\mathbf{s}_i))$ is indistinguishable from the uniform distribution on $\mathbb{Z}_q^{t+1}$, where the distribution is taken over uniform choices for $L(\mathbf{s}_i)$ and over uniformly sampled $\lambda_i \in \{0, 1\}$. Indeed, an extension of the left-over

hash lemma states that the distribution $(L(\mathbf{s}_1), \ldots, L(\mathbf{s}_t), w_1, \ldots, w_u)$, where $w_j = \sum_i \lambda_{j,i} L(\mathbf{s}_i) \pmod q$ for $1 \leq j \leq u$, is indistinguishable from uniform. Suppose, for a contradiction, that given a value $w \in \mathbb{Z}_q$ one could determine some information about possible values for $L(\mathbf{s}_i)$ such that $w = \sum_i \lambda_i L(\mathbf{s}_i) \pmod q$. Then one would have a distinguisher D that, on input $(l_1, \ldots, l_t, w)$ checks whether or not the values l_i are potentially consistent with the value w. The left-over hash lemma implies that no such distinguisher exists, and hence even if given u exact values $w_j = \sum_i \lambda_{j,i} L(\mathbf{s}_i) \pmod q$ one cannot learn anything about the values $L(\mathbf{s}_i)$. Since a decryption query returns even less information (just one bit of this value), it follows that there is no algorithm to learn $\mathbf{s}_i$ from decryption queries.

More precisely, if $t' \geq \log(q) + 2k$ then the statistical difference between the distribution on $\mathbb{Z}_q$ given by $\sum_{i=1}^{t'} \lambda_i L(\mathbf{s}_i)$ and the uniform distribution is at most 2^{-k}. The adversary does not even see the whole value, but only one bit of it. This means that the value output by the decryption oracle is indistinguishable from a uniform value. Since a uniform value is independent of the long-term secret key $\mathbf{t}_1, \ldots, \mathbf{t}_t$, it follows that the adversary *cannot learn a secret key from making queries of this form*.

To achieve security one can take $t \geq \log(q) + 3k$, but this is likely to be overkill in practice. Since q grows like n^L it is possible to satisfy this condition while also satisfying the necessary condition $t = O(\log(n))$ for Theorem 2. An open problem is to give a more precise analysis on distribution of a single bit of such a linear projection, and hence obtain a smaller value for t.

5.2 Adaptive Attack 2

Recall that the aim of this attack is to learn $\mathbf{e}$ and this is done by setting the I-th column of $\mathbf{C}$ to be the transpose of the j-th row $\mathbf{a}_j$ of $\mathbf{A}$ and setting all other columns of $\mathbf{C}$ to be zero. The difference in this MGSW case is that the value I varies, and is not known to the attacker.

First note that

$$\mathbf{a}_j = (\mathbf{b}_j \mathbf{t}_1 + e_{1,j}, \mathbf{b}_j \mathbf{t}_2 + e_{2,j}, \ldots, \mathbf{b}_j \mathbf{t}_t + e_{t,j}, \mathbf{b}_j)$$

and that the one-time secret key is $\mathbf{s}' = (\lambda_1, \lambda_2, \ldots, \lambda_t, \sum_i \lambda_i \mathbf{t}_i)^T$. Let I' be such that $q/4 < 2^{I'-1} \leq q/2$ and put $\mathbf{a}_j$ in the I'-th column of $\mathbf{C}$ and set all other columns to be zero. The attacker hopes that the decryption algorithm chooses $\lambda_1 = 1$ and $I = I'$. There is a $1/2$ probability that $\lambda_1 = 1$, and on average there will be about $t/2$ indices such that $\lambda_i = 1$. Hence, the probability that the decryption oracle chooses $I = I'$ is roughly $(1/2)(2/t) = 1/t$.

If $I = I'$ then the decryption oracle computes

$$\begin{aligned} \mathbf{C}_I^T \mathbf{s}' &= \mathbf{a}_j \mathbf{s}' \\ &= \sum_i \lambda_i (\mathbf{b}_j \mathbf{t}_i + e_{i,j}) - \mathbf{b}_j \sum_i \lambda_i \mathbf{t}_i \\ &= e_{1,j} + \sum_{i=2}^{t} \lambda_i e_{i,j} \pmod q. \end{aligned}$$

In other words, the attacker can "see" $e_{1,j}$ but with a noise term $E = \sum_{i=2}^{t} \lambda_i e_{i,j}$ added. Note that the $e_{i,j}$ terms are fixed and the distribution of the noise term E is over the choices of λ_i. Nevertheless, since the $e_{i,j}$ were originally sampled from a discrete Gaussian it follows that the mean value of E will be close to 0 and the distribution of E will be similar to a Gaussian. Hence, it is natural to hope that the E terms can be "averaged away" by making repeated decryption oracle queries.

More precisely, the attacker will choose an appropriate integer $-q/2 < u < q/2$ and call the decryption oracle on a matrix $\mathbf{C}$ whose I'-th column is the transpose of $\mathbf{a}'_j = \mathbf{a}_j + (u \mid 0, \cdots, 0)$. The decryption oracle (assuming it uses $I = I'$) therefore computes $e_{1,j} + u + E \pmod q$ where E is the noise term, and returns $\lfloor (e_{1,j} + u + E \pmod q)/2^{I-1} \rceil \rfloor$. The attacker therefore learns whether $|e_{1,j} + u + E| < 2^{I-2}$ or not. Repeating the same query (and again requiring that $I = I'$) will give the same computation but with a different value for the noise E. Hence, one should be able to learn the most significant bits of $e_{1,j}$ after sufficiently many queries.

Computing $e_{1,j}$ exactly using this approach is probably unrealistic, at the very least since the mean value of the distribution of the E is not known to an attacker. But once enough information about $\mathbf{e}_1$ is obtained by the attacker then they can finish the cryptanalysis by solving a much easier instance $(\mathbf{B}, \mathbf{b}' = \mathbf{B}\mathbf{t}_1 + \mathbf{e}')$ of LWE, where $\|\mathbf{e}'\| \ll \|\mathbf{e}_1\|$, using a lattice algorithm.

It is clear that this attack is much harder to mount than it was on the original scheme. So our multi-key system has provided some protection against adaptive attacks. Nevertheless, the attack has the potential to completely break the scheme and so we need to consider a different solution.

6 Dual Multiple Secret Scheme (DMGSW)

We now develop a dual version of the GSW scheme in the multi-secret-key case. Since we are using the dual scheme, the security now depends on the inhomogeneous short integer solution problem (ISIS) rather than LWE.

Definition 12. *(Inhomogeneous Short Integer Solution Problem (ISIS)) Let $q, n, m \in \mathbb{N}$ with $m > n$. Let χ be a distribution on $\mathbb{Z}$. Define the ISIS distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q^{n \times m}$ by*

$$(\mathbf{B}\mathbf{t} \pmod q, \mathbf{B})$$

where $\mathbf{B}$ is a uniformly chosen $n \times m$ matrix over $\mathbb{Z}_q$ and $\mathbf{t} \leftarrow \chi^m$ is an integer vector of length m.

The (decisional) (m, q, χ, n)-ISIS problem is to distinguish samples $(\mathbf{u}, \mathbf{B})$ taken from the ISIS distribution from samples taken from the uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q^{n \times m}$.

Ajtai [3] (also see Micciancio [22]) has shown that there exist distributions χ such that the (m, q, χ, n)-ISIS problem is hard for appropriately chosen parameters. Indeed, one can take χ to be a discrete Gaussian distribution or even the uniform distribution on $\{0, 1\}^m$ (see [24]).

Theorem 5. *Let $m > n \in \mathbb{N}$, let $q \in \mathbb{N}$ and let χ be a discrete Gaussian distribution on $\mathbb{Z}$ such that the (m, q, χ, n)-ISIS problem is hard. Let t be an integer such that $t = O(\log(m))$. Define two distributions $\mathcal{X}$ and $\mathcal{Y}$ as follows.*

- *$\mathcal{X}$ is the distribution on $n \times (t + m)$ matrices*

$$[\mathbf{u}_1|\cdots|\mathbf{u}_t|\mathbf{B}]$$

where $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$ is chosen uniformly at random and where, for all $1 \leq i \leq t$,

$$\mathbf{u}_i = \mathbf{B}\mathbf{t}_i \pmod q$$

where $\mathbf{t}_i$ is sampled from a discrete Gaussian distribution χ^m.
- *$\mathcal{Y}$ is the uniform distribution on $\mathbb{Z}_q^{n\times(t+m)}$.*

Then the two distributions $\mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable.

Proof. Let D be a probabilistic polynomial-time adversary that can distinguish $\mathcal{X}$ from $\mathcal{Y}$ with non-negligible advantage. For $1 \leq i \leq t + 1$ we introduce intermediate distributions $\mathcal{X}_i$ given by

$$[\mathbf{u}'_1|\cdots|\mathbf{u}'_{i-1}|\mathbf{u}_i|\cdots|\mathbf{u}_t|\mathbf{B}]$$

where $\mathbf{u}_i$ is as above and $\mathbf{u}'_i$ is uniformly chosen from $\mathbb{Z}_q^n$. Hence $\mathcal{X}_1 = \mathcal{X}$ and $\mathcal{X}_{t+1} = \mathcal{Y}$.

By assumption, D can distinguish $\mathcal{X}_1$ from $\mathcal{X}_{t+1}$ with noticeable advantage ϵ and so, by a standard hybrid argument, there is some i such that D can distinguish $\mathcal{X}_i$ from $\mathcal{X}_{i+1}$ with some noticeable advantage at least ϵ/t.

It is then easy to see that D gives an ISIS distinguisher: Given an ISIS challenge $(\mathbf{y}, \mathbf{B})$ one samples $\mathbf{u}'_1, \ldots, \mathbf{u}'_{i-1}$ uniformly and samples $\mathbf{u}_{i+1}, \ldots, \mathbf{u}_t$ from the ISIS distribution) and then calls D on

$$[\mathbf{u}'_1|\cdots|\mathbf{u}'_{i-1}|\mathbf{y}|\mathbf{u}_{i+1}|\cdots|\mathbf{u}_t|\mathbf{B}].$$

By assumption, no such distinguisher exists. Hence the theorem is proved. □

6.1 Dual MGSW Scheme (DMGSW Scheme)

We now describe our dual variant of the GSW13 scheme. Recall that the original GSW scheme has public key $\mathbf{A} = (\mathbf{B}\mathbf{t} + \mathbf{e}, \mathbf{B})$ based on LWE and ciphertext based on the ISIS-like problem $\mathbf{A}^T\mathbf{R}$. Just as with Regev's encyption scheme, the dual scheme has public key $(\mathbf{B}^T, \mathbf{B}^T\mathbf{T})$ based on ISIS and the ciphertext is based on the LWE-like instance $\mathbf{B}\mathbf{R} + \mathbf{X}$.

- $\mathsf{params} \leftarrow \mathsf{DMGSW.Setup}(1^k, 1^L)$:
  1. Choose a modulus $q = q(k)$, lattice dimension parameter $m = m(k, L)$, parameter n and distribution $\chi = \chi(k, L)$, appropriately chosen in order to achieve at least 2^k security against known ISIS attacks;
  2. Let $l = \lfloor \log q \rfloor + 1$ and $N = (t + m)l$ and output $\mathsf{params} = (m, q, \chi, n)$.

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{DMGSW.KeyGen}(\mathsf{params})$:
  - Sample $\mathbf{t}_i^T = (t_{i,1}, \cdots, t_{i,n})$ from χ^m and compute $\mathbf{e}_i = (\mathbf{I}_i \mid -\mathbf{t}_i^T)^T$ where $\mathbf{I}_i$ is the ith row of the $t \times t$ identity matrix;
  - Sample uniformly a matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ and compute $\mathbf{u}_i = \mathbf{B}\mathbf{t}_i$ for $1 \leq i \leq t$. Set $\mathbf{A} = [\mathbf{u}_1 \mid \cdots \mid \mathbf{u}_t \mid \mathbf{B}] \in \mathbb{Z}_q^{n \times (t+m)}$. Note that $\mathbf{A}\mathbf{e}_i = 0$;
  - Output the public key $\mathbf{A}$ and the private key $(\mathbf{e}_1, \ldots, \mathbf{e}_t)$.
- $\mathbf{C} \leftarrow \mathsf{DMGSW.Enc}(\mathsf{params}, \mathsf{pk}, \mu)$ where $\mu \in \{0, 1\}$:
  1. Sample a uniform matrix $\mathbf{R} \leftarrow \mathbb{Z}_q^{n \times N}$ and $\mathbf{X} \leftarrow \chi^{(t+m) \times N}$;
  2. Compute $\mathbf{C} = \mu\mathbf{G} + \mathbf{A}^T\mathbf{R} + \mathbf{X} \pmod q \in \mathbb{Z}_q^{(t+m) \times N}$ where $\mathbf{G}$ is the $(t + m) \times N$ gadget matrix;
  3. Return $\mathbf{C}$.

  Remark 3. If we used the original GSW formulation then we would have written $\mathbf{C}$ as

$$\begin{aligned} &\mathsf{Flatten}(\mu\mathbf{I} + \mathsf{BitDecomp}(\mathbf{A}^T\mathbf{R} + \mathbf{X})) \\ =\ &\mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\mu\mathbf{I}) + \mathbf{A}^T\mathbf{R} + \mathbf{X}) \\ =\ &\mathsf{BitDecomp}(\mu\mathbf{G} + \mathbf{A}^T\mathbf{R} + \mathbf{X}) \pmod q. \end{aligned}$$

- $\mu' \leftarrow \mathsf{DMGSW.Dec}(\mathsf{params}, \mathsf{sk}_i, \mathbf{C})$:
  1. Choose $\lambda_1, \cdots, \lambda_t$ in $\mathbb{Z}$ such that they are not all zero and generate a one-time secret key $\mathbf{e}' = \sum_{i=1}^t \lambda_i \mathbf{e}_i$ such that $\|\mathbf{e}'\|$ is short. Note that $\mathbf{A}\mathbf{e}' \equiv 0 \pmod q$.
     There are several ways one might choose the λ_i. One approach would be to sample them uniformly from $\{0, 1\}$. This leads to good values for $\|\mathbf{e}'\|$ and it is the approach used in our analysis. Another approach would be to choose them from a discrete Gaussian distribution on $\mathbb{Z}$ with small standard deviation, and this might lead to a higher security scheme (see discussion in Section 7). Finally, one could apply some form of rejection sampling to the resulting vectors $\mathbf{e}'$, in order to control their size and also the extent to which they leak information about the vectors $\mathbf{e}_i$.
  2. Determine an integer $1 \leq I = (i-1)l + j \leq tl$ such that $\lambda_i = 1$ and $2^{j-1} \in (q/4, q/2]$;
  3. Let $\mathbf{C}_I$ be the I-th column of $\mathbf{C}$ and compute $u = \langle \mathbf{C}_I, \mathbf{e}' \rangle = \mathbf{C}_I^T\mathbf{e}' \pmod q$. Note that

$$\begin{aligned} \mathbf{C}^T\mathbf{e}' &= \mu\mathbf{G}^T\mathbf{e}' + \mathbf{R}^T\mathbf{A}'\mathbf{e}' + \mathbf{X}^T\mathbf{e}' \\ &= \mu\mathbf{G}^T\mathbf{e}' + \mathbf{X}^T\mathbf{e}'. \end{aligned}$$

     Hence

$$\mathbf{C}_I^T\mathbf{e}' \ = \ \mu(0, 0, \ldots, 0, 2^{j-1}, 0, \ldots 0)\mathbf{e}' + E \ = \ \mu\lambda_i 2^{j-1} + E$$

     where $E = \mathbf{X}_I^T\mathbf{e}'$ is a small noise term.
  4. Return $\lfloor u/2^{j-1} \rceil \in \{0, 1\}$.

- $\mathsf{DMGSW.Eval}(\mathsf{params}, \mathbf{C}_1, \cdots, \mathbf{C}_l)$: The homomorphic operations are exactly the same as in the original scheme.
  - $\mathsf{DMGSW.Add}(\mathbf{C}_1, \mathbf{C}_2)$: output

$$\mathbf{C}_1 + \mathbf{C}_2 = (\mu_1 + \mu_2)\mathbf{G} + \mathbf{A}^T(\mathbf{R}_1 + \mathbf{R}_2) + (\mathbf{X}_1 + \mathbf{X}_2) \in \mathbb{Z}_q^{(t+m)\times N};$$

  - $\mathsf{DMGSW.Mult}(\mathbf{C}_1, \mathbf{C}_2)$: Compute the $N \times N$ matrix $\mathbf{G}^{-1}(\mathbf{C}_2)$ and compute and output $\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$.
    Note that

$$\begin{aligned}\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2) &= \big(\mu_1\mathbf{G} + \mathbf{A}^T\mathbf{R}_1 + \mathbf{X}_1\big)\mathbf{G}^{-1}(\mathbf{C}_2)\\ &= \mu_1\mathbf{C}_2 + \mathbf{A}^T\mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mathbf{X}_1\mathbf{G}^{-1}(\mathbf{C}_2)\\ &= \mu_1\mu_2\mathbf{G} + \big(\mathbf{A}^T\mathbf{R}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1\mathbf{A}^T\mathbf{R}_2\big)\\ &\quad + \underbrace{\big(\mathbf{X}_1\mathbf{G}^{-1}(\mathbf{C}_2) + \mu_1\mathbf{X}_2\big)}_{error^{Mult}} \in \mathbb{Z}_q^{(t+m)\times N}.\end{aligned}$$

  - $\mathsf{DMGSW}.NAND(\mathbf{C}_1, \mathbf{C}_2)$: Compute $\mathbf{G}^{-1}(\mathbf{C}_2)$ and output $\mathbf{G} - \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$.

6.2 Correctness

In this subsection, we determine parameters for the correctness of decryption and homomorphic operations. In this section we assume $\mathbf{e}' = \sum_{i=1}^{t} \lambda_i \mathbf{e}_i$ is formed by sampling $\lambda_i \in \{0, 1\}$ uniformly. Recall that the $\mathbf{e}_i$ have entries chosen from a discrete Gaussian distribution χ with standard deviation σ which is B-bounded for some integer such as $B = 6\sigma$. We also have $\|\mathbf{e}_i\| \leq 2\sqrt{m}\sigma$ with overwhelming probability. Hence the entries of $\mathbf{e}'$ are bounded by $B' = tB$. It also follows that $\|\mathbf{e}'\| \leq 2\sqrt{tm}\sigma$.

We say that a ciphertext $\mathbf{C}$ is at *level* 0 if it is the output of the encryption algorithm on some message. We say that a ciphertext $\mathbf{C}$ is at *level* $i \geq 1$ if it has been formed by running the Evaluate algorithm at most i times level 0 ciphertexts.

Definition 13. *We say that a ciphertext* $\mathbf{C} \in \mathbb{Z}_q^{(t+m)\times N}$ *is* E-noisy *if*

$$\mathbf{C}^T\mathbf{e}' = \mu\mathbf{G}^T\mathbf{e}' + \mathbf{X}^T\mathbf{e}'$$

where $\|\mathbf{X}^T\mathbf{e}'\|_\infty \leq E$.

Note that if $\mathbf{C}$ is an E-noisy ciphertext with $E < q/8$ then decryption works correctly as $\mathbf{C}_I^T\mathbf{e}' \equiv \mu 2^{j-1} + e \pmod q$ with $|e| \leq E < q/8 < 2^{j-2}$ and so

$$\frac{\mathbf{C}_I^T\mathbf{e}' \pmod q}{2^{j-1}} = \mu + \tfrac{e}{2^{j-1}} = \mu + \epsilon$$

where $-\frac{1}{2} < \epsilon < \frac{1}{2}$.

Lemma 4. *Let* χ *be a* B*-bounded distribution on* $\mathbb{Z}$*. Suppose* $E \geq tB + mB^2$*. Then a ciphertext at level* 0 *is* E*-noisy.*

Proof. We have $\mathbf{C} == \mu\mathbf{G} + \mathbf{A}^T\mathbf{R} + \mathbf{X}$ where $\mathbf{X}$ is sampled from $\chi^{(t+m)\times N}$. Write $\mathbf{X}$ as $[\frac{\mathbf{X}^\#}{\mathbf{X}^*}]$ where $\mathbf{X}^\# \in \mathbb{Z}^{t\times N}$ and $\mathbf{X}^* \in \mathbb{Z}^{m\times N}$. Recall that $\mathbf{e}' = (\lambda_1, \ldots, \lambda_t \mid -\sum_i \lambda_i \mathbf{t}_i)^T$.

We therefore have

$$\mathbf{X}^T\mathbf{e}' = (\mathbf{X}^\#)^T(\lambda_1, \ldots, \lambda_t)^T - \sum_i \lambda_i (\mathbf{X}^*)^T\mathbf{t}_i$$

and we want to bound the I-th entry of this (coming from the I-th row of $\mathbf{X}^T$, which is the transpose of the I-th column $\mathbf{X}_I$ of $\mathbf{X}$).

Since $\|\mathbf{X}_I^*\|_2 \leq 2\sqrt{m}\sigma$ and $\|\mathbf{t}_i\|_2 \leq 2\sqrt{m}\sigma$ the Cauchy-Schwarz inequality shows that

$$|(\mathbf{X}_I^*)^T\mathbf{t}_i| \leq 4m\sigma^2 \leq mB^2.$$

We also have $|(\mathbf{X}_I^\#)^T(\lambda_1, \ldots, \lambda_t)^T| \leq tB$. So the error is bounded in infinity norm by $tB + mB^2$. □

Now we turn our attention to homomorphic operations. The idea is to ensure that a level i ciphertext is $(N+1)^i E$-noisy where E is the level zero noise, and so if $(N+1)^L E \leq q/8$ then decryption is correct even after performing L levels of operations.

Lemma 5. *Let notation and parameters be as above (in particular, $E \geq tB + mB^2$). Let $\mathbf{C}$ be any ciphertext at level $i \leq L$. Then $\mathbf{C}$ is $(N+1)^i E$-noisy.*

Proof. Let $\mathbf{C}_1$ and $\mathbf{C}_2$ be two ciphertexts that are noisy encryptions of $\mu_1, \mu_2 \in \{0, 1\}$ and such that $(\mathbf{C}_i)^T\mathbf{e}' = \mu_i\mathbf{G}^T\mathbf{e}' + E_i$ for $i = 1, 2$.

Consider adding the two ciphertexts. Suppose both ciphertexts have level i and their noise is bounded by $(N+1)^i E$ in the infinity norm. We compute $\mathbf{C}^{Add} = \mathbf{C}_1 + \mathbf{C}_2$. Note that

$$(\mathbf{C}^{Add})^T\mathbf{e}' = (\mu_1 + \mu_2)\mathbf{G}^T\mathbf{e}' + (E_1 + E_2)$$

and $\|E_1 + E_2\|_\infty \leq \|E_1\|_\infty + \|E_2\|_\infty \leq 2(N+1)^i E \leq (N+1)^{i+1}E$. So the claim is true in this case.

Now consider multiplication. We compute $\mathbf{C}^{Mult} = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$. Note that

$$(\mathbf{C}^{Mult})^T\mathbf{e}' = \mu_1\mu_2\mathbf{G}^T\mathbf{e}' + \mathbf{G}^{-1}(\mathbf{C}_2)^T E_1 + \mu_2 E_2$$

and

$$\|\mathbf{G}^{-1}(\mathbf{C}_2)^T E_1 + \mu_2 E_2\|_\infty \leq \|\mathbf{G}^{-1}(\mathbf{C}_2)^T E_1\|_\infty + \|\mu_2 E_2\|_\infty \leq N\|E_1\|_\infty + \|E_2\|_\infty.$$

The result follows. The same calculation holds for NAND gates. □

Suppose that $q/8 > (N+1)^L E$ and consider the evaluation of a Boolean circuit of depth L consisting of NAND gates. The input is level zero ciphertexts that are E-noisy. We have shown that at each level the noise is multiplied by a factor of at most $(N+1)$,. Therefore the error terms in the final ciphertext have norm bounded by $\big(N+1\big)^L E$ and decryption works correctly.

6.3 Security of the Dual Scheme

The security of the DMGSW scheme is based on the ISIS and LWE assumptions. We now show that the DMGSW scheme is IND-CPA-secure under the ISIS assumption by using Theorem 5 to show that the scheme is secure.

Theorem 6. *The* DMGSW *scheme with* $t = O(\log(n))$ *is IND-CPA-secure under the* (m, q, χ, n)*-ISIS assumption and the LWE assumption.*

Proof. (Sketch) The proof of security consists of two steps, which can be represented as game hops. The first game hop is to replace the public key with a uniformly chosen matrix $\mathbf{A}$. The second game hop is to replace the ciphertext with a uniformly chosen matrix $\mathbf{C}$. It is immediate that an adversary cannot have non-negligible advantage in the IND-CPA game when $\mathbf{C}$ is a uniformly chosen matrix, since then $\mathbf{C}$ is independent of the message μ. It is necessary to show that the adversary behaviour is the same across the game hops.

- **Hybrid one:** In this game we replace the public key with a uniform matrix. Encryption of the challenge ciphertext is the same as in the scheme. If the success probability of the adversary has non-neglible difference between these two games then the adversary is an algorithm that can distinguish the ISIS instance of the real public key from a uniformly chosen matrix.
  By the ISIS assumption and Theorem 5 no such algorithm exists.
- **Hybrid two:** In this game we replace the ciphertext $\mathbf{C}$ by a uniformly sampled matrix in $\mathbb{Z}_q^{(t+m)\times N}$. If the success probability of the adversary in this game is noticeably different from its success probability in Hybrid 1 then we have a distinguisher between the LWE distribution $\mathbf{A}^T\mathbf{R} + \mathbf{X} \pmod q$ and uniform. By the LWE assumption there is no such polynomial-time distinguisher.

Finally, the game in Hybrid 2 is independent on the message bit μ, and so the adversary can have zero advantage in this game. □

7 Security of the Dual Multi-Secret GSW Scheme Against Adaptive Attacks

The main observation is that there are no error terms, so Adaptive Attack 2 does not occur. However, we still have to worry about a version of Attack 1. We will show that a similar argument to that given in Section 5.1 can be used to show that known attacks do not apply to the dual scheme.

The arguments rely on a Gaussian version of the leftover hash lemma, rather than the uniform case used earlier. The following theorem is a special case of Theorem 2 of Agrawal, Gentry, Halevi and Sahai [2].

Theorem 7. *(One Dimensional Leftover Hash Lemma) Let* $\epsilon, \sigma \in \mathbb{R}$ *be such that* $\epsilon > 0$ *and* $\sigma > C$ *for some absolute constant (see [2]). Let* $t \geq 10\log(8t^{1.5}\sigma)$ *and* $s' \geq 4t\log(1/\epsilon)$*. Then the statistical distance between the following two distributions is bounded by* 2ϵ*.*

– *Choose a length t vector $\mathbf{X} \in \mathbb{Z}^t$ with entries chosen from the discrete Gaussian distribution on $\mathbb{Z}^t$ with parameter σ and a length t vector $\mathbf{z} \in \mathbb{Z}^t$ with entries chosen from a discrete Gaussian distribution on $\mathbb{Z}^t$ with parameter s' and compute and output $\mathbf{X}^T\mathbf{z}$.*
– *Choose and output an element from the discrete Gaussian distribution on $\mathbb{Z}$ with parameter $\sigma s'$.*

Recall that the one-time secret key is

$$\mathbf{e}' = \sum_{i=1}^{t} \lambda_i \mathbf{e}_i = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_t \\ \\ -\sum_{i=1}^{t} \lambda_i \mathbf{t}_i \end{pmatrix} \in \mathbb{Z}_q^{t+m}$$

The decryption algorithm computes $\langle \mathbf{C}_I, \mathbf{e}' \rangle = \mathbf{C}_I^T \mathbf{e}' \pmod q$. Hence we still have to imagine an attacker who inserts a chosen vector into the I-th column of the matrix $\mathbf{C}$ and makes a decryption query. As with the attack in Section 5.1, the adversary does not know I, but can guess this value with high probability. Indeed, with probability $1/2$ we have $\lambda_1 = 1$ and, by setting all other columns of $\mathbf{C}$ to be zero, one can ensure that the decryption oracle only returns a non-zero value when $\lambda_1 = 1$.

We use the same formulation as introduced in Section 5.1. Hence we write $L : \mathbb{Z}_q^{t+m} \to \mathbb{Z}_q$ for the linear map corresponding to multiplication by $\mathbf{C}_I^T$. An attacker will obtain one bit of

$$L(\mathbf{e}') \;=\; \sum_{i=1}^{t} \lambda_i L(\mathbf{e}_i) \;=\; \sum_{i=1}^{t} \lambda_i L((\mathbf{I}_i, -\mathbf{t}_i^T)^T).$$

As in Section 5.1 we can argue that almost all of the vectors $(\mathbf{I}_i, -\mathbf{t}_i^T)^T$ are not in the kernel of L. There are two cases to consider, depending on whether L preserves "shortness" of vectors.

The first case is when the values $L(\mathbf{e}_i)$ behave like uniformly chosen entries in $\mathbb{Z}_q$. In this case, the analysis from Section 5.1 based on the left-over-hash lemma (and an assumption that L is surjective and does not vanish on most of the secret keys) is sufficient to deduce that the information on $L(\mathbf{e}')$ is independent of the values $L(\mathbf{e}_i)$ and so an adversay cannot learn the private key from queries of this form.

The second case occurs if, for example, L is a projection $L((w_1, \ldots, w_{t+m})^T) = w_i$ onto a single coordinate $t < i \leq t+m$. In this case the values $L(\mathbf{e}_i)$ will be samples from a discrete Gaussian and one cannot use the left-over-hash lemma to justify that $L(\mathbf{e}')$ carries no information about the $L(\mathbf{e}_i)$. Instead, for this case we apply Theorem 7. In this case we need to assume the λ_i are sampled themselves from a discrete Gaussian distribution χ with parameter s. Suppose an attacker sees $\sum_{i=1}^{t} \lambda_i L(\mathbf{e}_i)$ where the $L(\mathbf{e}_i)$ are sampled independently from the discrete Gaussian distribution χ. Then Theorem 7 states that for $t = O(\log(m))$ (this is because $\sigma = O(\sqrt{m})$ for the hardness of LWE and so $t = O(\log(\sigma)) = O(\log(m))$) then this integer is indistinguishable from a sample from a discrete Gaussian with parameter σ^2. Now, in our application the $L(\mathbf{e}_i)$

are fixed rather than independently sampled, but are also not provided to the adversary. Instead, the adversary view is simply $\sum_{i=1}^{t} \lambda_i L(\mathbf{e}_i)$ with distribution inherited from $(\lambda_1, \ldots, \lambda_t) \in \chi^t$, but the Gaussian left-over-hash lemma still gives an assurance that this value is independent of the $L(\mathbf{e}_i)$. Further, as explained earlier, the attacker only sees one bit rather than the entire value $\sum_{i=1}^{t} \lambda_i L(\mathbf{e}_i)$, and so we have good evidence that the scheme resists attacks of this type.

8 Conclusion

We have given two variants of the GSW13 scheme and discussed adaptive attacks on them. The most secure scheme is a multi-key version of the dual GSW scheme, and we give evidence why this scheme should resist adaptive attacks.

Acknowledgments

The authors would like to thank Damien Stehlé and the anonymous reviewers for their helpful advice and comments. This work was supported by the National Natural Science Foundation of China (No.61472097), Specialized Research Fund for the Doctoral Program of Higher Education of China (No.20132304110017) and International Exchange Program of Harbin Engineering University for Innovation-oriented Talents Cultivation.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Advances in Cryptology–EUROCRYPT 2010, pp. 553–572. Springer (2010)
2. Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete gaussian leftover hash lemma over infinite domains. In: Advances in Cryptology-ASIACRYPT 2013, pp. 97–116. Springer (2013)
3. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Miller, G.L. (ed.) Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing. pp. 99–108. ACM (1996)
4. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. In: Advances in Cryptology–CRYPTO 2014, pp. 297–314. Springer (2014)
5. Biasse, J.F., Fieker, C.: Subexponential class group and unit group computation in large degree number fields. LMS Journal of Computation and Mathematics 17(A), 385–403 (2014)
6. Biasse, J.F., Song, F.: Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 893–902. SIAM (2016)
7. Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1. In: Advances in Cryptology–CRYPTO 98. pp. 1–12. Springer (1998)
8. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Advances in Cryptology–CRYPTO 2012, pp. 868–886. Springer (2012)
9. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. Electronic Colloquium on Computational Complexity (ECCC) 18, 111 (2011)
10. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. pp. 309–325. ACM (2012)

11. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science. pp. 97–106. IEEE (2011)
12. Chenal, M., Tang, Q.: On key recovery attacks against existing somewhat homomorphic encryption schemes. In: Progress in Cryptology-LATINCRYPT 2014, pp. 239–258. Springer (2014)
13. Chenal, M., Tang, Q.: Key recovery attacks against NTRU-based somewhat homomorphic encryption schemes. In: Information Security, pp. 397–418. Springer (2015)
14. Cramer, R., Ducas, L., Peikert, C., Regev, O.: Recovering short generators of principal ideals in cyclotomic rings. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 559–585. Springer (2016)
15. Dahab, R., Galbraith, S., Morais, E.: Adaptive key recovery attacks on NTRU-based somewhat homomorphic encryption schemes. In: Information Theoretic Security, pp. 283–296. Springer (2015)
16. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 169–169. ACM Press (2009)
17. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. pp. 197–206. ACM (2008)
18. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Advances in Cryptology–CRYPTO 2013, pp. 75–92. Springer (2013)
19. Impagliazzo, R., Levin, L.A., Luby, M.: Pseudo-random generation from one-way functions. In: Proceedings of the twenty-first annual ACM symposium on Theory of computing. pp. 12–24. ACM (1989)
20. Loftus, J., May, A., Smart, N.P., Vercauteren, F.: On CCA-secure somewhat homomorphic encryption. In: International Workshop on Selected Areas in Cryptography. pp. 55–72. Springer (2011)
21. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Advances in Cryptology–EUROCRYPT 2012, pp. 738–755. Springer (2012)
22. Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In: 43rd Symposium on Foundations of Computer Science (FOCS 2002). pp. 356–365. IEEE Computer Society (2002)
23. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Advances in Cryptology–EUROCRYPT 2012, pp. 700–718. Springer (2012)
24. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 21–39. Springer (2013)
25. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key fhe. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 735–763. Springer (2016)
26. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 333–342. ACM (2009)
27. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. SIAM Journal on Computing 40(6), 1803–1844 (2011)
28. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing. pp. 84–93. ACM (2005)

29. Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: International Workshop on Public Key Cryptography. pp. 420–443. Springer (2010)
30. Zhang, Z., Plantard, T., Susilo, W.: On the CCA-1 security of somewhat homomorphic encryption over the integers. In: International Conference on Information Security Practice and Experience. pp. 353–368. Springer (2012)