This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

Towards Multi-Hop Homomorphic Identity-Based Proxy Re-Encryption via Branching Program

Zengpeng Li, Chunguang Ma, Ding Wang

Abstract—Identity-based Proxy re-encryption (IBPRE) is a powerful cryptographic tool for various applications such as access control system, secure data sharing, and secure e-mail forwarding. Most of existing efficient IBPRE schemes are based on the Diffie-Hellman assumption, and they only focus on single-hop construction. Based on the work of Chandran et al.'s lattice-based proxy re-encryption (PRE) scheme (PKC'14) and Yamada's lattice-based identity-based encryption (IBE) scheme (EUROCRYPT'16), in this paper we first show the possibility of assembling lattice-based IBE into lattice-based PRE. Then we present the construction of a new efficient single-hop homomorphic IBPRE from learning with errors (LWE) via key homomorphic computation. Furthermore, using branching program (BP), we obtain an efficient multi-hop IBPRE scheme. To the best of our knowledge, our scheme is the first multi-hop homomorphic IBPRE scheme via BP. Our scheme supports homomorphic evaluation and is proved secure under the decisional LWE assumption.

Index Terms—Multi-Hop; Identity-Based Proxy Re-Encryption; Key Switching Mechanism; Homomorphic Evaluation

1 Introduction

Consider a scenario in which a proxy server wants to achieve once ciphertext transformation, i.e., converts ciphertexts of Alice (i.e., delegator) to ciphertexts of Bob (i.e., delegatee). Then the transformation ciphertexts can be decrypted by Bob's secret key. This scenario is called the single-hop proxy re-encryption (hereafter 1-hop PRE) scheme. Moreover, if there exists another user, Eve, then, upon receiving the ciphertext from Alice, Bob re-encrypts the ciphertext and sends it to Eve. In this setting, Eve decrypts the ciphertext from Bob using her secret key. This scenario is called 2-hop PRE. Repeating this procedure many times, we can obtain multi-hop PRE. Actually, the above scenario can be viewed as the PRE under identity-based encryption (IBE) setting. In order to achieve fine-grained access control, the identity-based proxy re-encryption (hereafter IBPRE, a.k.a., PRE under IBE setting) came into being. Nowadays, IBPRE schemes play a crucial role in the foundation of privacy preserving, hence various information security systems were proposed to protect users' privacy [1], [2], [3] etc.

We observe that, most PRE or IBPRE schemes and optimizations were developed by using pairing-based cryptography (i.e., under the Decisional Diffie-Hellman (DDH) assumption). However, with the advancement of the quantum computer, DDH-based cryptography cannot prevent the quantum attacks. Thus post-quantum cryptography is now receiving increasing emphasis. Lattice-based cryptography is one of the typical representatives of post-quantum cryptography. The worst-case hardness of lattice problems (such as the Short Integer Solution problem (SIS) and the Learning with Errors (LWE) problem [4], [5]) have been proved to be a phenomenal success in fully homomorphic encryption (FHE) [6]. Importantly, lattice-based cryptography quickly caught up with pairing-based cryptography. In this case, many cryptography primitives were re-constructed by using lattice-based cryptography, such as IBE schemes [7], attribution-based encryption (ABE), PRE scheme and optimizations [8], [9].

Although the possibility of LWE-based PRE constructions was shown by Xagawa in his thesis [8] and many follow-ups lattice-based PRE constructions were proposed (e.g., [9], [10], [11], [12] etc.), there nevertheless remains one primitive for which lattice-based PRE scheme is still far behind: Identity-based Proxy Re-Encryption (IBPRE) [13] from lattice.

To the best of our knowledge, there is only one scheme of lattice-based IBPRE, given by Singh, Rangan, and Banerjee (SRB) [14], and no subsequent work. We note that, the main drawback of SRB's construction is that they only focus on single-hop IBPRE and cannot propose the multi-hop construction. This limitation is a serious obstacle and prevents to capture the real-world multi-hop requirements, i.e., multi-hop IBPRE. Moreover, there was only one trapdoor for finding short vectors in SRB's construction. Concretely, they just used $\mathbf{G}$ trapdoor function to enable the real IBPRE system to generate the master secret key. However, no other trapdoor was provided to enable the simulator to generate short vectors. Apparently, the multi-hop constructions under paring-based setting don't imply the multi-hop constructions under lattice-based setting. Hence, we view it as an important question to determine

whether we can achieve an efficient multi-hop IBPRE *scheme in a lattice-based setting?*

To solve this issue, we first obtain an efficient lattice-based IBPRE construction by following the methodology of Yamada's IBE scheme. Then, we achieve multi-hop IBPRE by introducing the Branching Program (BP). The

- *Z.P. Li and C.G. Ma are with the College of Computer Sciences and Technology, Harbin Engineering University, Harbin 150001, P.R. China; and State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. (Email: {lizengpeng,machungaung}@hrbeu.edu.cn)*
- *D. Wang is with the School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, P.R. China (Email: wangdingg@pku.edu.cn;).*

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

crux of this issue is how to assemble IBE scheme into the PRE scheme efficiently and how to embed the IBPRE into BP. Below, we sketch our main technique.

1.1 Our Contribution and Technical Overview

In the following, we describe our technical ideas in a high level, and give a detailed description in Section 3.

- Our first contribution is that we develop a variant of key switching procedure by using Gentry-Peikert-Vaikuntanathan's (GPV) [5] scheme. More concretely, as far as we know, using Brakerski et al. [15] key-switching technique, Chandran et al. [11] proposed two types of single-hop PRE schemes. The first one is based on Regev's scheme [4] and the second one is based on GPV (a.k.a., dual-Regev) [5] scheme. In this paper, we use the dual-Regev scheme as the building block to design the IBPRE scheme. In this setting, we need the key-switching mechanism to achieve re-encryption algorithm. However, there does not exist any GPV-style (or dual-Regev-style) key-switching mechanism. Inspired by the errorless key-switching mechanism of Li et al. [16], we first tweak their errorless key-switching mechanism, and then develop a GPV-style key-switching mechanism.
- Our second contribution is that we construct the first IBPRE scheme based on the lattice. The crux of lattice-based IBPRE is how to assemble IBE scheme into the PRE scheme efficiently. Concretely, inspired by the work of Yamada [7], we first use a hash function $\mathcal{H}(\cdot)$ to map each entry of id to a matrix and we can obtain many matrices; secondly, we utilize homomorphic multiplication to restructure these matrices and obtain $\mathcal{H}(id)$; thirdly, we sample a public matrix (e.g., $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$) to generate a matrix $(\mathbf{A}|\mathcal{H}(id))$. For future convenience, the matrix $(\mathbf{A}|\mathcal{H}(id))$ is called *user-specific* matrix. In this case, there exists an identity public matrix, e.g., $(\mathbf{u}|\mathbf{A}|\mathcal{H}(id))$, for each identity id, where $\mathbf{u}$ is a fixed vector such that $(\mathbf{A}|\mathcal{H}(id))\mathbf{e} = \mathbf{u} \pmod q$ for secret key $\mathbf{e}$. Next, we follow GPV-style encryption algorithm, use the matrix $(\mathbf{u}|\mathbf{A}|\mathcal{H}(id))$ to encrypt the message and obtain the ciphertext of identity id. In order to achieve the proxy re-encryption, we transform the ciphertext of identity $id^{(i)}$ into the ciphertext of identity $id^{(j)}$ by using the above "*GPV-style key-switching mechanism*" for $j \geq i+1$. In this way, we can naturally assemble IBE scheme into GPV-style PRE and obtain a lattice-based IBPRE scheme. We describe our construction for IBPRE in Section 3 and Section 4.
- Our third contribution is that we obtain the multi-hop homomorphic IBPRE scheme via Branching Program (BP). The point is that how to use BP to achieve multi-hop IBPRE. Concretely, the core of BP is NAND gate which can be expressed by the form of $\mathbf{X}\cdot\mathbf{G}^{-1}(\mathbf{Y})$ for some specified matrices $\mathbf{X}$ and $\mathbf{Y}$. Actually, this structure of $\mathbf{X}\cdot\mathbf{G}^{-1}(\mathbf{Y})$ is similar to the homomorphic multiplication $\mathbf{C}_1\cdot\mathbf{G}^{-1}(\mathbf{C}_2)$ for the two ciphertexts $\mathbf{C}_1$ and $\mathbf{C}_2$, and it's also similar to the re-encryption algorithm $\mathbf{C}_j := \mathbf{K}\cdot\mathbf{G}^{-1}(\mathbf{C}_i)$ of PRE for the re-encryption key $\mathbf{K}$ and the ciphertext at input side $\mathbf{C}_i$. Thus, in order to obtain multi-hop IBPRE, we need our multi-hop IBPRE scheme support homomorphic operation. In this setting, we first tweak the encryption algorithm of IBPRE and make it support homomorphic operation. Then, we embed the re-encryption key and ciphertext into the NAND gate and transform the NAND gate into BP. We remark that, Chandran et al. [11] have presented a multi-hop PRE scheme via the ideal circuit. However, their scheme only allows limited times hop and we cannot find the concrete construction. Fan et al. [17] also constructed a multi-hop PRE scheme, however, their scheme is based on G-trapdoor function introduced by [18]. To the best of our knowledge, NC1 circuit is one type of ideal circuit and by Barrington's theorem, any NC1 circuit can be converted into the polynomial-size BP with constant-width 5 and polynomial-length 4^d for circuit depth d. Moreover, BP can be used to compute encrypted data [19] or represent the decryption circuit. Most importantly, before running the BP to achieve the multi-hop re-encryptions, the input values (the re-encryption keys and the ciphertexts) don't need to be known in advance. That's the reason why we develop a multi-hop homomorphic IBPRE scheme via BP.

1.2 Other Related Work

In the following, we describe some related work.

Proxy Re-Encryption: The concept of PRE was first proposed by Blaze et al. [20] at EUROCRYPT'98. Their construction is based on the ElGamal encryption scheme [21] and chosen-plaintext attack (CPA) secure under the decisional Diffie-Hellman (DDH) assumption. Along this line, many follow-up PRE schemes (e.g., [22], [23], [24] etc.) and optimizations were constructed under DDH assumption. The possibility of LWE-based PRE constructions was shown by Xagawa in his thesis [8], but the scheme lacks a complete security analysis. Subsequently, Aono et al. [9] proposed the first scheme at INDOCRYPT'13, which is key-private LWE-based PRE, but their scheme is weakly key-private. A major breakthrough in LWE-based PRE arrived with the work of Kirshanova [10] and Chandran et al. [11] at PKC'14. Kirshanova [10] proposed a new unidirectional LWE-based PRE scheme which is collusion-safe and does not require any trusted authority for the re-encryption key generation and at the same time, Chandran et al. [11] constructed an obfuscator re-encryption, functional re-encryption, and multi-hop re-encryption from the decisional LWE assumption, without going through FHE [6] respectively. Very recently, utilizing re-encryption techniques of Gentry's FHE [6], Nishimaki and Xagawa [12] proposed two types of key-private LWE-based PRE scheme based on Regev [4] scheme and Linder-Peikert [25] scheme respectively. Actually, Chandran et

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

al. [11] and Aono et al. [9] also took the "convert-then-rerandomize" approach to construct PRE scheme. Moreover, Ma et al. [26] gave a variant of single-hop PRE scheme by using the encryption algorithm of Gentry-Sahai-Waters (GSW) [27] to encrypt the message, and got a new single-hop homomorphic PRE scheme that supports homomorphic operation.

Identity-Based Encryption: The concept of IBE was first proposed by Boneh and Franklin [28]. The construction of Boneh-Franklin's IBE is based on pairing under DDH assumption and cannot prevent quantum attacks. Most notably, the existing IBE constructions (e.g., [29], [30], [31], [7] etc.) from lattice follow the general blueprint of constructing the lattice-based IBE scheme first introduced by Agrawal, Boneh, and Boyen at EUROCRYPT'10 [29]. Along this line, a breakthrough in fully key-homomorphic encryption technique arrived with the work of Boneh et al. [32], which was inspired by the work of FHE [27] and they proposed the fully key-homomorphic technique to embed the entries of identity into the circuit. Followed by Yamada [7], he constructed an adaptively secure LWE-based IBE, which captures the security notions of real work.

Identity-Based Proxy Re-Encryption: Green and Ateniese [13] proposed the first (IBPRE) scheme based on Boneh-Franklin's IBE scheme [28] in the random oracle model at ACNS'07. Since Green and Ateniese [13] showed the possibility of IBPRE constructions under DDH assumption, many IBPRE schemes with the desirable properties were proposed. More specially, Ren et al. [33] gave the chosen-ciphertext attack (CCA) secure hierarchical IBPRE scheme. Shao et al. [34] gave the CCA-secure multi-hop hierarchical IBPRE scheme. Moreover, Liang et al. [35] constructed an attribute-based PRE scheme, etc. Note that the above schemes are based on DDH assumption.

We remark that, to the best of our knowledge, our scheme is the first lattice-based multi-hop homomorphic IBPRE scheme via BP in the standard model and computes the identity matrix via fully key homomorphic technique, which corresponds to practical needs. For comparison sake, we give a comparison result in the Table 1. We stress that there are many follow-up works and optimizations. We just compare some related works with our scheme:

Application: On the other hand, in cloud computing, in order to protect users' privacy and maintain the confidentiality of sensitive data, the cryptographers use cryptography approaches to design various cryptosystems [1], [2], [3], [37], etc. Notably, the main applications of IBPRE (or PRE) are access control on cloud storage, secure data sharing, secure e-mail forwarding, etc [36]. Wang et al. [38] proposed a novel proxy-oriented data uploading and remote data integrity checking model in identity-based proxy public key cryptography which is based on the technique of PRE.

1.3 Paper organization

The remainder of this paper is organized as follows. In Section 2 we formally define the LWE assumption and present notation that will be used throughout the paper. In Section 3 we describe our single-hop homomorphic IBPRE scheme. In Section 4 we describe our multi-hop homomorphic IBPRE scheme. Finally, in Section 5, we give a conclusion.

2 Preliminaries

In this section we introduce required notations, definitions and lemmas which are taken from previous works.

2.1 Notation

For $n \in \mathbb{N}$, we let $[n]$ denote 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. We denote vector $\mathbf{x}$ via bold lower-case letter and matrix $\mathbf{A}$ via bold upper-case letter. We use ":=" to denote deterministic assignment. Let $\mathcal{D}_{\mathbb{Z}_q^m,\sigma}$ be the truncated discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter σ, that is, we replace the output by $\mathbf{0}$ whether the $\|\cdot\|_\infty$ norm exceeds $\sqrt{m}\cdot\sigma$. Note that $\mathcal{D}_{\mathbb{Z}^m,\sigma}$ is $\sqrt{m}\cdot\sigma$-bounded.

Norms. We will give well-known norms that will be used in the following sections. ℓ_∞ norm is: $||\mathbf{v}||_\infty = \max\{|v_1|, \cdots, |v_n|\}$ and Euclidean norm is: $||\mathbf{v}||_2 = \sqrt{\sum_{i=1}^{n} |v_i|^2}$. For matrix $\mathbf{A} \in \mathbb{Z}^{k\times m}$, let $\widetilde{\mathbf{A}}$ be the result of applying Gram-Schmidt(GS) orthogonalization to the columns of $\mathbf{A}$, s.t., $\|\mathbf{A}\|_{GS} = \|\widetilde{\mathbf{A}}\|$. $||\mathbf{A}||$ denotes the l_2 length of the longest column of $\mathbf{A}$. (For notational convenience, we denote the length of a matrix is the norm of its longest column: $||\mathbf{A}|| = \max_i ||\mathbf{a}_i||$.)

Indistinguishability. Moreover, for a random variable X and an element x we use $\Pr[X = x]$ to denote the probability that X outputs x.

Definition 2.1 (Statistical Distance)**.** *Let X_0 and X_1 be two random variables with range D (i.e., a finite set). We call $\delta(X_0, X_1) := \frac{1}{2}\sum_{d\in D} |\Pr[X_0 = d] - \Pr[X_1 = d]|$ the statistical distance between X_0 and X_1.*

Definition 2.2 (Statistical Indistinguishability)**.** *We say that X_0 and X_1 are statistically indistinguishable (a.k.a., statistically close), written $X_0 \approx_s X_1$, if $\delta(X_0(\lambda), X_1(\lambda))$ is negligible in λ.*

Similarly, let $X_0 = \{X_0(\lambda)\}_{\lambda\in\mathbb{N}}$ be families of variables. We say that X_0 and X_1 are computationally indistinguishable and write $X_0 \approx_c X_1$ if there is no PPT distinguishers which can distinguish the above variables.

Injective Map. We denote an efficiently computable injective map (i.e., one of the canonical map) S that maps a bit string $\mathsf{ID} \in \{0,1\}^\kappa$ to a subset $S(\mathsf{ID})$ of $[1, \xi]^d$, where $\xi = \lceil \kappa^{1/d} \rceil$ and d is some integer.

Definition 2.3 ([39] Def2.1)**.** *A distribution ensemble $\chi = \chi(\lambda)$ over the integers is called B-bounded (denoted $|\chi| \le B$) if there exists:*

$$\Pr_{x \xleftarrow{\$} \chi}[|x| \ge B] \le 2^{-\tilde{\Omega}(n)}$$

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

Table 1
Comparison of LWE-based PRE and IBPRE Schemes

Scheme	Assumption	Security	Hop	Direction	Bit	IBE	FHE
Blaze et al. [20]	DDH	CPA	Single	Bi-	Multi	✗	✗
Green et al. [13]	DDH	CPA	Single	Uni-	Multi	✓	✗
Matsuo [36]	DDH	CPA	Single	Uni-	Multi	✓	✗
Ren et al. [33]	DDH	CCA	Single	Uni-	Multi	✓	✗
Shao et al. [34]	DDH	CCA	Multi	Uni-	Multi	✓	✗
Xagawa [8]	LWE	NONE	Single	Uni-	Multi	✗	✗
Aono et al. [9]	LWE	CPA	Single	Uni-	Multi	✗	✗
Chandran et al. [11]	LWE	CPA	Multi	Uni-	Multi	✗	✗
Kirshanova [10]	LWE	CCA	Single	Uni-	Multi	✗	✗
Singh et al. [14]	LWE	CPA	Single	Uni-	Single	✓	✗
Nishimaki et al. [12]	LWE	CPA	Single	Uni-	Multi	✗	✗
Fan et al. [17]	LWE	CCA	Multi	Uni-	Multi	✗	✗
Ma et al. [26]	LWE	CPA	Single	Uni-	Multi	✗	✓
Our scheme	LWE	CPA	Multi	Uni-	Single	✓	✓

- Bi-: Bi-Direction;
- Uni-: Uni-Direction;
- ✗: IBE/HE setting is not achieved;
- ✓: IBE/HE setting is achieved;

Lemma 2.4 ([5] Lemma 2.9). *For any n-dimension lattice Λ, $c \in span(\Lambda)$, real $\varepsilon \in (0,1)$, and gaussian parameter $r \geq \eta_\varepsilon(\Lambda)$:*

$$\Pr_{\mathbf{x} \leftarrow D_{\Lambda,r,\mathbf{c}}}\left[||\mathbf{x}-\mathbf{c}|| > r\cdot\sqrt{n}\right] \leq \frac{1+\varepsilon}{1-\varepsilon}\cdot 2^{-n}$$

The final fact we need for certain applications is an upper bound on the probability of the mode (the most likely element) of a discrete Gaussian; equivalently, it is a lower bound on the min-entropy of the distribution.

2.2 Lattice Background and Learning with Errors

Lemma 2.5 (Matrix-vector Leftover Hash Lemma (LHL), [15] Lemma 2.1). *Let $\lambda \in \mathbb{Z}$, n, $q \in \mathbb{N}$, $m \geq n\log q + 2\lambda$, $\mathbf{r} \xleftarrow{R} \{0,1\}^m$ and $\mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n$. Sample a uniform random matrix $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{m\times n}$, then the statistical distance between the distributions $(\mathbf{A}, \mathbf{A}^T\mathbf{r})$ and $(\mathbf{A}, \mathbf{y})$ is as follows:*

$$\Delta\big((\mathbf{A}, \mathbf{A}^T\cdot\mathbf{r}), (\mathbf{A}, \mathbf{y})\big) \leq 2^{-\lambda} \tag{2.1}$$

Lemma 2.6 ([40] Lemma 4.4). *1) For $\forall\ k > 0$, $\Pr[|e| > k\cdot\sigma, e \leftarrow \mathcal{D}_\sigma^1] \leq 2\cdot\exp(-\frac{k^2}{2})$;*
2) For $\forall\ k > 0$, $\Pr[\|\mathbf{e}\| > k\cdot\sigma\cdot\sqrt{m}, \mathbf{e} \leftarrow \mathcal{D}_\sigma^m] \leq k^m\cdot\exp(\frac{m}{2}\cdot(1-k^2))$.

Remark 2.7. *Throughout the paper, we suppose $\sigma \geq 2\sqrt{n}$. Therefore, if $\mathbf{e} \leftarrow \mathcal{D}_\sigma^m$ then we have, on average, that $\|\mathbf{e}\| \approx \sqrt{m}\cdot\sigma$. Lemma 2.6 (2) implies that $\|\mathbf{e}\| \leq 2\sigma\sqrt{m}$ with overwhelming probability.*

Lemma 2.8 ([29] Lemma 12). *Let vector $\mathbf{x}$ be some vector in $\mathbb{Z}^m$ and let $\mathbf{e} \leftarrow D_{\mathbb{Z}^m,r}$. Then the quantity $|\ \mathbf{x}^T\cdot\mathbf{e}\ |$ when treated as an integer in $[0,\cdots,q-1]$ satisfies*

$$|\ \mathbf{x}^T\cdot\mathbf{e}\ | \leq ||\mathbf{x}||r\omega(\sqrt{log\ m}) + ||\mathbf{x}||\sqrt{m}/2$$

with all but negligible probability in m. Where r is a gaussian parameter and defined in Lemma 2.4.

We define the decisional version as follows,

Definition 2.9 (Decision-LWE$_{n,q,\chi,m}$). *Assume given an independent sample $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times 1}$, where the sample is distributed according to either: (1) $\mathcal{A}_{\mathbf{s},\chi}$ for a uniform random $\mathbf{s} \in \mathbb{Z}_q^n$ (i.e., $\{(\mathbf{A}, \mathbf{b}) : \mathbf{A} \leftarrow \mathbb{Z}_q^{m\times n}, \mathbf{s} \leftarrow \mathbb{Z}_q^{n\times 1}, \mathbf{e} \leftarrow \chi^{m\times 1}, \mathbf{b} = \mathbf{A}\cdot\mathbf{s}+\mathbf{e} \pmod q\}$), or (2) the uniform distribution (i.e., $\{(\mathbf{A}, \mathbf{b}) : \mathbf{A} \leftarrow \mathbb{Z}_q^{m\times n}, \mathbf{b} \leftarrow \mathbb{Z}_q^{m\times 1}\}$). Then, the above two distributions are computationally indistinguishable.*

Remark 2.10. *Regev and others [4], [41], [42], [43] show that the LWE assumption is at least as hard as solving the worst-case Shortest Independent Vectors Problem (SIVP) for polynomial approximation factors. We omit the corollary of these schemes' results. More details will be found in [4], [41], [42], [43].*

2.3 Trapdoor

Below, we show two sampling algorithms of SampleLeft and SampleRight [29] which will be used in our scheme.

Lemma 2.11 ([5], [29]). *There exist some parameters $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{B} \in \mathbb{Z}_q^{n\times n}$, $\mathbf{R} \in \mathbb{Z}_q^{m\times n}$, a vector $\mathbf{u} \in \mathbb{Z}_q^{n\times 1}$, two short basis $\mathbf{T_A}$ of $\Lambda_q^\perp(\mathbf{A})$ and $\mathbf{T_B}$ of $\Lambda_q^\perp(\mathbf{B})$, and a Gaussian parameter s, then*

- SampleLeft$(\mathbf{A}, \mathbf{B}, \mathbf{T_A}, \mathbf{u}, s)$: *Takes as input $\mathbf{A}$, $\mathbf{B}$, $\mathbf{T_A}$, $\mathbf{u}$ and s, then outputs a vector $\mathbf{e} \leftarrow \Lambda_q^{\mathbf{u}}(\mathbf{F})$ for $\mathbf{F} = (\mathbf{A}|\mathbf{B})$, where $\mathbf{e}$ is statistically close to $D_{\Lambda_q^{\mathbf{u}}(\mathbf{F}),s}$.*
- SampleRight$(\mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{T_B}, \mathbf{u}, s)$: *Takes as input $\mathbf{A}$, $\mathbf{B}$, $\mathbf{R}$, $\mathbf{T_B}$, $\mathbf{u}$ and s, then outputs a vector $\mathbf{e} \leftarrow \Lambda_q^{\mathbf{u}}(\mathbf{F})$ for $\mathbf{F} = (\mathbf{A}|\mathbf{AR}+\mathbf{B})$, where $\mathbf{e}$ is statistically close to $D_{\Lambda_q^{\mathbf{u}}(\mathbf{F}),s}$.*

2.4 Basic Tools

We use the same techniques proposed by Brakerski et al. [44], [39]. First fix $q, m \in \mathbb{N}$ and let $\ell_q = \lfloor\log q\rfloor + 1$ and $N = m\cdot\ell_q$.

Lemma 2.12 ([18]). *For any $N \geq m\lceil\log q\rceil$ there exist a computable gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{m\times N}$ and an efficiently computable deterministic inverse (a.k.a., "short preimage") function $\mathbf{G}^{-1}(\cdot)$. The inverse function $\mathbf{G}^{-1}(\mathbf{M})$ takes as*

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

input a matrix $\mathbf{M} \in \mathbb{Z}_q^{m \times m'}$ *for any* m' *and outputs a matrix* $\mathbf{G}^{-1}(\mathbf{M}) \in \{0,1\}^{N \times m'}$ *such that* $\mathbf{G}\mathbf{G}^{-1}(\mathbf{M}) = \mathbf{M}$.

Remark 2.13. *For future convenience, the gadget matrix* $\mathbf{G}$ *from Micciancio and Peikert [18] can be expressed by* $\mathbf{G} = \mathbf{I}_{m\times m} \otimes \mathbf{g} \in \mathbb{Z}_q^{m \times N}$ *where* $\mathbf{g} = (1, 2, 4, \ldots, 2^{\ell_q - 1})^T$. *For* $\mathbf{v} \in \mathbb{Z}_q^m$ *we have* $\mathsf{PowerOf2}(\mathbf{v}) = \mathbf{v}^T\mathbf{G}$. *For* $\mathbf{v} \in \mathbb{Z}_q^N$ *we have* $\mathsf{Bit}^{-1}(\mathbf{v}) = \mathbf{G}\mathbf{v}$. *For* $\mathbf{a} \in \mathbb{Z}_q^m$ *the algorithm* $\mathsf{Bit}(\mathbf{a})$ *can be renamed as* $\mathbf{G}^{-1}(\mathbf{a})$.

Remark 2.14. *We set* $\mathbf{t}_i$ *as the* i*-th unit vector* $(0, \cdots, 1, \cdots, 0)^T$. *Consider the gadget matrix over* $\mathbb{Z}_q^{m\times N}$ *with the following view*

$$\begin{aligned}\mathbf{G} &= (2^0, 2^1, \cdots, 2^{\ell_q - 1}) \otimes \mathbf{I}_{m\times m} \\ &= [2^0\mathbf{t}_1, \cdots, 2^{\ell_q-1}\mathbf{t}_1, \cdots, 2^0\mathbf{t}_N, \cdots, 2^{\ell_q-1}\mathbf{t}_N] \\ &= [\mathbf{G}[1], \cdots, \mathbf{G}[N]]\end{aligned}$$

where any column vector $\mathbf{G}[i]$ *over* $\mathbb{Z}_q^{m\times 1}$ *for* $i \in \{0, \cdots, m \cdot \ell_q - 1\}$, *the structure of* $\mathbf{G}[i]$ *should be* $2^j \cdot \mathbf{t}_i = (0, \cdots, 2^j, \cdots, 0)^T$, *where* $j := i \mod \ell_q$. *For example, without loss of generality, the* $(\ell_q - 1)$*-th vector is* $\mathbf{G}[\ell_q - 1] := 2^{(\ell_q-1)} \cdot \mathbf{t}_1 = (2^{(\ell_q-1)}, 0, \cdots, 0)^T$.

2.5 Predecessor Function for Circuit

Below is the definition of Predecessor Function for Circuit $\mathsf{Pred}_\Psi(i)$ and its corollary for on-the-fly branching programs BPOTF, which are from Brakerski et al. [45].

Definition 2.15 (Predecessor Function for Circuit)**.** *We first denote a circuit as in Barington Theorem by* Ψ. *Then we denote the predecessor function of* Ψ *by* $\mathsf{Pred}_\Psi(i)$ *where* i *is the index of length. Lastly, we set the label of* $\mathsf{Pred}_\Psi(i)$*'s output gate to be* 0 *and we label* $\mathsf{Pred}_\Psi(i)$*'s input gate by the index of the variable. For example, set a label* i *for a gate,* $\mathsf{Pred}_\Psi(i)$ *returns* (j_1, j_2) *which are the labels of the wires feeding this gate.*

Corollary 2.16 (Barrington On-The-Fly)**.** *Given access to a* $\mathsf{Pred}_\Psi(i)$ *of a depth* d *circuit, there exists a uniform machine* BPOTF *that outputs the layers* $(p_{0,t}, p_{1,t})$ *of the branching program from Barrington Theorem for* BPOTF*'s width* $t \in [L]$. *Each layer takes time* $O(d)$ *to produce, and the total space used by* BPOTF *is* $O(d)$.

2.6 Branching Program

Below, we describe the computational model of permutation "*branching programs*" (BP). We note a corollary from Barrington construction, which allows computing the BP "on-the-fly", layer by layer, keeping only small state. Hence, we define the BP similarly to [46], [45].

Definition 2.17. *A permutation* BP $\prod$ *with* ℓ *variables, width* k *and length* L *is a sequence of* L *tuples* $(p_{0,t}, p_{1,t})_{t\in[L]}$ *which is called the instruction. For a function* $var : [L] \to [\ell]$, *each tuple is composed of a pair of permutations* $p_{0,t}$, $p_{1,t} : [k] \to [k]$. *The* BP *takes as input a binary vector* $\mathbf{x} = (x_1, \cdots, x_\ell) \in (0,1)^\ell$, *and outputs a bit* $b \in \{0,1\}$. *The execution of* $\prod$ *is as follows:*

- *The program keeps a state integer* $s \in \{0, \cdots, k\}$, *initially* $s_0 = 1$;
- *On every step* $t = 1, \cdots, L$, *the next state is determined (recursively) using the* t*-th instruction:* $s_t := p_{x_{var(t)}}(s_{t-1})$.

All in all, $s_t := p_{0,t}(s_{t-1})$ *if* $x_{var(t)} = 0$, *and otherwise* $s_t := p_{1,t}(s_{t-1})$. *Finally, after the* L*-th iteration, the* BP *outputs 1 if and only if* $s_L = 1$.

Remark 2.18. *In this paper, we would like to evaluate a* BP *homomorphically. Thus, we prefer to represent an integer state* $s \in \{1, 2, 3, 4, 5\}$ *by a 0-1 state vector* $\mathbf{v} \in \{0,1\}^5$. *The computation then processes as follows: The idea is that* $\mathbf{v}_t[i] = 1 \Leftrightarrow s_t = i$. *We initialize* $\mathbf{v}_0[1] = 1$ *and* $\mathbf{v}_0[i] = 0$ *for* $i \in \{2, 3, 4, 5\}$ *and evaluate using the following recursive formula:* $\mathbf{v}_t[i] = 1 \Leftrightarrow p_{t,x_{var(t)}}(s_{t-1}) = i$. *Turning this around, for every* $1 \le i \le 5$, $\mathbf{v}_t[i] = 1$ *if and only if:*

- *either* $\mathbf{v}_{t-1}[p_{t,0}^{-1}(i)] = 1$ *and* $x_{var(t)} = 0$
- *or* $\mathbf{v}_{t-1}[p_{t,1}^{-1}(i)] = 1$ *and* $x_{var(t)} = 1$.

Hence, equivalently, we have the following formula:

$$\begin{aligned}\mathbf{v}_t[i] &= \mathbf{v}_{t-1}[p_{t,0}^{-1}(i)](1 - x_{vat(t)}) + \mathbf{v}_{t-1}[p_{t,1}^{-1}(i)]x_{vat(t)} \\ &= \mathbf{v}_{t-1}[\gamma_{t,i,0}](1 - x_{vat(t)}) + \mathbf{v}_{t-1}[\gamma_{t,i,1}]x_{vat(t)} \quad (2.2)\end{aligned}$$

where $\gamma_{t,i,0} \triangleq p_{t,0}^{-1}(i)$ *and* $\gamma_{t,i,1} \triangleq p_{t,1}^{-1}(i)$ *are constant parameters that are publicly computable given the description of the* BP. *After the* L*-th iteration, we accept if and only if* $s_L = 1$, *that is we output* $\mathbf{v}_L[1]$. *Please keep in mind the form that we will use it in our homomorphic evaluation.*

Below, we recall the Barrington's Theorem.

Theorem 2.19 (Barrington's Theorem [47])**.** *Every Boolean* NAND *circuit* C *that takes* ℓ *inputs and has depth* d *can be computed by a width-5 permutation* BP *of length* 4^d. *Given the circuit* C*'s description and the* BP *'s description* $\prod$ *can be computed in* $\mathrm{poly}(\ell, 4^d)$.

2.7 Identity-Based Proxy Re-Encryption

Definition 2.20. *An identity-based proxy re-encryption (*IBPRE*) scheme is a tuple of probabilistic polynomial time (*PPT*) algorithms as follows,*

$$(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{ReKeyGen}, \mathsf{ReEnc})$$

we denote $\mathcal{ID}$ *as the identity space of* IBPRE *scheme. Moreover, if there exists a collision resistant hash function (e.g.,* $\mathcal{H}$*):* $\{0,1\}^* \to \mathcal{ID}$, *then we can use an arbitrary string as an identity. In more detail:*

- $(mpk, msk) \leftarrow \mathsf{IBPRE.Setup}(1^\lambda)$: *Takes the security parameters* λ *as input and outputs a master public key* mpk *and a master secret key* msk;
- $sk_{id} \leftarrow \mathsf{IBPRE.KeyGen}(mpk, msk, id)$*: Takes the* mpk, msk, *and an identity* $id \in \mathcal{ID}$ *as input, then it outputs a private key* sk_{id}, *where* id *is implicitly included in* sk_{id};
- $\mathbf{c}_{id} \leftarrow \mathsf{IBPRE.Enc}(mpk, id, \mu)$*: Takes an* mpk, *an identity* $id \in \mathcal{ID}$, *a message* μ *and* pk_{id} *corresponding to* id, *then it outputs a ciphertext* $\mathbf{c}_{id}$;
- $rk^{(i\to j)} \leftarrow \mathsf{IBPRE.ReKeyGen}(mpk, id^{(i)}, id^{(j)}, sk_{id^{(i)}})$: *In order to generate re-encryption key* $rk^{(i\to j)}$ *from user* i *to user* j, *takes as input* mpk, $sk_{id^{(i)}}$ *under* $id^{(i)}$ *for* i, *the identity* $id^{(i)}$ *and* $id^{(j)}$ *for user* i *and* j;

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

- $\mathbf{c}_{id^{(j)}} \leftarrow \mathsf{IBPRE.ReEnc}(\mathbf{c}_{id^{(i)}}, rk^{(i\to j)})$: *In order to transform user i's ciphertext $\mathbf{c}_{id^{(i)}}$ to user j's ciphertext $\mathbf{c}_{id^{(j)}}$ (also called user i's re-encryption ciphertext), takes as input $\mathbf{c}_{id^{(i)}}$ and $rk^{(i\to j)}$. Notably, for convenience, hereafter, we write $\mathbf{c}_{id^{(j)}}$ as $\mathbf{c}_j$;*
- $\mu \leftarrow \mathsf{IBPRE.Dec}(mpk, sk_{id}, \mathbf{c}_{id})$: *Takes the mpk, sk_{id}, and $\mathbf{c}$, then it outputs the message μ when the ciphertext is in a valid form, otherwise, outputs $\perp$.*

Correctness: For all n, all $id \in \mathcal{ID}$, and all μ in the specified message space,

$$\Pr[\mathsf{Dec}(mpk, sk_{id}, \mathsf{Enc}(mpk, id, \mu)) = \mu] = 1 - \mathrm{negl}(\lambda)$$

holds, where the probability is taken over the randomness used in $(mpk, msk) \leftarrow \mathsf{Setup}(1^\lambda)$, $sk_{id} \leftarrow \mathsf{KeyGen}(mpk, msk, id)$, and $\mathsf{Enc}(mpk, id, \mu)$.

Security: The adversary $\mathcal{A}$ is allowed to adaptively choose the IBPRE secret key queries and re-encryption queries. This security notion is defined by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

1) At the outset of the game, $\mathcal{C}$ runs the $\mathsf{Setup}(1^\lambda) \to (mpk, msk)$ and sends mpk to $\mathcal{A}$;
2) $\mathcal{A}$ adaptively makes the key-extraction queries, and it first submits $id \in \mathcal{ID}$ to $\mathcal{C}$, more specially, $\mathcal{A}$ first submits the form of $(extract, id \in \mathcal{ID})$ to $\mathcal{C}$, then $\mathcal{C}$ obtains a secret key $sk_{id} \leftarrow \mathsf{KeyGen}(\mathsf{params}, mpk, msk, id)$ for identity i and sends the sk_{id} to $\mathcal{A}$, then $\mathcal{A}$ adds id to the honest user set $\mathcal{ID}$. Here $\mathcal{A}$ can repeat the query many times;
3) $\mathcal{A}$ issues the re-encryption key query $rk^{(i\to j)}$ for identities $id^{(i)}$ and $id^{(j)}$ where $i \neq j$, then the challenger obtains $rk^{(i\to j)}$ via the re-key generation algorithm $\mathsf{ReKeyGen}(mpk, (id^{(i)}, id^{(j)}), sk_{id^{(i)}})$ and sends back $rk^{(i\to j)}$ to $\mathcal{A}$; (Here we must stress that we do not allow re-encryption key generation queries between a corrupted and an uncorrupted party, i.e. we require that the queries occur at either all of (i, j), $i \neq j$ are honest parties, or alternatively all are corrupted parties.)
4) $\mathcal{A}$ submits query $(id^{(i)}, id^{(j)}, \mathbf{c}_{id^{(i)}})$, $i \neq j$ to "*Re-encryption Oracle*", the challenger $\mathcal{C}$ in turn generates ciphertext $\mathbf{c}_{id^{(j)}} \leftarrow \mathsf{ReEnc}(rk^{(i\to j)}, \mathbf{c}_{id^{(i)}})$;
5) (**Challenge Phase.**) At some point, $\mathcal{A}$ submits identity $id^* \in \mathcal{ID}$ and messages m_0, m_1 to "*Challenge oracle*", then $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and returns $\mathbf{c}_{id_{i^*}} \leftarrow \mathsf{Enc}(mpk, id_{i^*}, m_b)$ to $\mathcal{A}$. Actually, after the challenge query, $\mathcal{A}$ may continue to make key-extraction queries, with the added restriction that $id \neq id^*$;
6) (**Guess**.) This is done only once, $\mathcal{A}$ finally outputs $b' \in \{0, 1\}$ as a guess of b, if $b' = b$, then outputs 1, otherwise, outputs 0. Here, the id^* is the challenge identity, and $id_1, \cdots, id_Q$ are identities for which $\mathcal{A}$ has made key-extraction queries.

We denote the advantage of an adversary as $|\Pr[b' = b] - \frac{1}{2}|$. Then we say the uni-directional IBPRE scheme is IND-ID-CPA-secure if there is no PPT adversaries have an overwhelming advantage.

Remark 2.21. *To the best of our knowledge, we can easily observe that the above security definition is equivalent to the definition of* IND-ID-*CPA/CCA defined by Boneh and Franklin at CRYPTO'01 [28]. The only difference is that, in our* IBPRE *scheme, the adversary $\mathcal{A}$ makes some queries of the form* "Re-encryption key generation" *and* "Re-encryption oracle". *Hence, if $rk^{(i\to j)}$ is not deployed, then the* IBPRE *remains* IND-ID-*CPA/CCA-secure.*

Proposition 2.22 (Single/Multi-Hop IBPRE)**.** *We say the* IBPRE *scheme is single-hop if the number of the hops between the output side and the input side is $j - i = 1$, which only allows the proxy server to re-encrypt the ciphertexts one time. If $j - i = L > 1$, we say the* IBPRE *scheme is L-hop (a.k.a., multi-hop* IBPRE*), which allows the proxy server to re-encrypt the ciphertexts up to L times.*

Compared with original IBPRE, the following definition of the homomorphic IBPRE has the same PPT algorithms except that the encryption algorithm supports homomorphic evaluation.

Definition 2.23 (Homomorphic IBPRE)**.** *A scheme as the definition of* IBPRE *is called* homomorphic IBPRE *scheme if it has the following evaluation algorithm:*

- $\mathbf{c}^\star \leftarrow \mathsf{Eval}(mpk, C, id, (\mathbf{c}_1, \cdots, \mathbf{c}_n))$ *Takes as input an mpk, an identity id, a list of ciphertexts $\mathbf{c}_1, \cdots, \mathbf{c}_n$, and a circuit $C \in \mathcal{C}_\lambda$, and it outputs a ciphertext $\mathbf{c}^\star$, where n is polynomial over λ.*

meanwhile, with the following properties,

- **Correctness of Re-Encrypted Ciphertexts:**

$$\Pr[\mathsf{Dec}\Big(sk_{id}, \mathsf{ReEnc}\big(rk, \mathsf{Enc}(mpk, id, \mu)\big)\Big) := \mu]$$

  with overwhelming probability.
- **Homomorphisms:** *The property of homomorphic is preserved as long as the ciphertexts in operations are under the same public key, for the encryption and re-encryption algorithms. In more detail, for any λ, any $\mu_1, \cdots, \mu_\ell \in \{0, 1\}^*$, and $C \in \mathcal{C}_\lambda$, we have that*

$$\begin{aligned}&\mathcal{C}(\mu_1, \cdots, \mu_n)\\ &= \mathsf{Dec}\Big(sk_{id}, id, \big(\mathsf{Eval}\big(mpk, id, (C,\\ &\mathsf{ReEnc}(rk, id, \mathbf{c}_1), \cdots, \mathsf{ReEnc}(rk, id, \mathbf{c}_n))\big)\big)\Big).\end{aligned}$$

- **Security:** *The property of security is the same as the security definition of* IBPRE*. We denote by the IND-ID-CPA game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Hence, we omit the description of the CPA game. So, if the advantage of* PPT *adversary $\mathcal{A}$ is negligible in λ, $\mathbf{Adv}_{\mathcal{A}} = |\Pr[b' = b] - \frac{1}{2}|$, then we say the homomorphic* IBPRE *scheme is IND-ID-CPA-secure.*

3 Single-Hop IBPRE Scheme

In this section, we first give a detailed description of GPV-style key switching, which is an important tool used in our construction. Then we present our main construction of IBPRE by the fully key homomorphic technique of Boneh et al. [32]. Lastly, we use the GPV-style key switching to achieve re-encryption. More details are as follows:

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

3.1 Yet Another Key Switching via Dual Regev

Our GPV-style key switching technique is greatly influenced by LWE-based key switching construction of Brakerski and Vaikuntanathan [15], [48]. But their construction is based on Regev [4] scheme, ours is based on Dual Regev [5] scheme. Most notably, Chandran et al. [11] also constructed key switching which is based on Dual Regev [5] scheme, but they created switching-key by a trapdoor, e.g., sampling algorithm $\mathrm{SampleD}(\cdot)$, which incurs an expensive overhead. In this paper, inspired by the work of Li et al. [16], we improve the key-switching mechanism via Dual Regev [5] scheme, which is followed by the framework of [15], [48].

Before describing our GPV-style key switching mechanism, we review the GPV (a.k.a., dual Regev) scheme first. In dual Regev scheme, the key generation algorithm generates public key $\mathbf{P} := (\mathbf{u}, \mathbf{B}) \in \mathbb{Z}_q^{m\times 1} \times \mathbb{Z}_q^{m\times n}$, where $\mathbf{B} \in \mathbb{Z}_q^{m\times n}$ is a public matrix and $\mathbf{u} = \mathbf{B}\cdot\mathbf{e} \pmod q$ for secret key $\mathbf{e} \in \chi^{n\times 1}$. In order to encrypt message $\mu \in \{0,1\}$, the encrypter computes $\mathbf{c} = \mathbf{P}^T\cdot\mathbf{r} + \frac{q}{2}(\mu|\mathbf{0}^n)^T + \mathbf{x} \pmod q$, where $\mathbf{r} \leftarrow \mathbb{Z}_q^{n\times 1}$ and $\mathbf{x} \leftarrow \chi^{(n+1)\times 1}$. In this setting, the decrypter decrypts ciphertexts $\mathbf{c}$ by computing $\langle \mathbf{c}, [1, -\mathbf{e}]^T\rangle$. Below we give a detailed description of the important "GPV-style key switching" technique in our construction.

- $\mathbf{P}_{sk_I:sk_O} \leftarrow \mathsf{SwitchKeyGen}(sk_I, sk_O)$:
  1) For the "input" and "output" secret key $sk_I = [1, -\mathbf{e}_I^T]^T \in \mathbb{Z}_q^{n_I\times 1}$ and $sk_O = [1, -\mathbf{e}_O^T]^T \in \mathbb{Z}_q^{n_O\times 1}$, set $sk_I^T\cdot\mathbf{G} := \mathsf{PowerOf2}_q(sk_I) \in \mathbb{Z}_q^{1\times \hat{n}_I}$, where $\hat{n}_I = n_I \times \lfloor \log q \rfloor$ and the gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{n_I\times \hat{n}_I}$.
  2) Sample a random matrix $\mathbf{A}_{I:O} \leftarrow \mathbb{Z}_q^{\hat{n}_I\times(n_O-1)}$ and an random vector $\mathbf{e}_\chi \leftarrow \chi_q^{\hat{n}_I\times 1}$, then compute $\mathbf{u}_{I:O} = \mathbf{A}_{I:O}\cdot\mathbf{e}_O \in \mathbb{Z}_q^{\hat{n}_I\times 1}$, then $\mathbf{b}_{I:O} = \mathbf{A}_{I:O}\cdot\mathbf{e}_O + \big([1, -\mathbf{e}_I^T]\cdot\mathbf{G}\big)^T + \mathbf{e}_\chi \in \mathbb{Z}_q^{\hat{n}_I\times 1}$.
  3) Output $\mathbf{P}_{I:O} = [\mathbf{b}_{I:O} \mid \mathbf{A}_{I:O}] \in \mathbb{Z}_q^{\hat{n}_I\times n_O}$.
- $\mathbf{c}_O \leftarrow \mathsf{SwitchKey}(\mathbf{P}_{I:O}, \mathbf{G}^{-1}(\mathbf{c}_I))$:
  1) To switch a ciphertext from sk_I to sk_O, compute "output" ciphertext $\mathbf{c}_O := \mathbf{P}_{I:O}^T\cdot\mathbf{G}^{-1}(\mathbf{c}_I) \in \mathbb{Z}_q^{n_O\times 1}$, where $\mathbf{P}_{I:O}\cdot[1, -\mathbf{e}_O^T]^T = \mathbf{b}_{I:O} - \mathbf{A}_{I:O}\cdot\mathbf{e}_O := \big([1, -\mathbf{e}_I^T]\cdot\mathbf{G}\big)^T + \mathbf{e}_\chi \in \mathbb{Z}_q^{\hat{n}_I\times 1}$ with $\mathbf{G}^{-1}(\mathbf{c}_I) := \mathsf{Bit}(\mathbf{c}_I) \in \mathbb{Z}_q^{\hat{n}_I\times 1}$.

Notably, there is no noise element $\langle \mathsf{Bit}_q(\mathbf{c}_I), \mathbf{error}_O\rangle$ in Eq.(3.1) compared with the [15]. Obliviously, it will improve computation efficiency of key switching.

Lemma 3.1 (Correctness). *Set $sk_I \in \mathbb{Z}^{n_I}$, $sk_O \in \mathbb{Z}^{n_O}$ and $\mathbf{c}_I \in \mathbb{Z}_q^{n_I}$ be any vectors. Suppose there exist $\mathbf{P}_{I:O} \leftarrow \mathsf{SwitchKeyGen}(sk_I, sk_O)$ and $\mathbf{c}_O \leftarrow \mathsf{SwitchKey}(\mathbf{P}_{I:O}, \mathbf{c}_I)$, then $\langle \mathbf{c}_O, sk_O\rangle = \langle \mathbf{c}_I, sk_I\rangle + \big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T\mathbf{e}_\chi \pmod q$.*

Proof: Considering $\mathbf{c}_O := \mathbf{P}_{I:O}^T\cdot\mathbf{G}^{-1}(\mathbf{c}_I) \in \mathbb{Z}_q^{n_O\times 1}$ and $sk_O = [1, -\mathbf{e}_O^T]^T \in \mathbb{Z}_q^{n_O\times 1}$, we hold that

$$\begin{aligned}\langle \mathbf{c}_O, sk_O\rangle &= \big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T\cdot\underline{\mathbf{P}_{I:O}\cdot\big([1, -\mathbf{e}_O^T]^T\big)} \\ &= \langle \mathbf{c}_I, \big([1, -\mathbf{e}_I^T]^T\big)\rangle + \big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T\cdot\mathbf{e}_\chi \\ &= \langle \mathbf{c}_I, sk_I\rangle + \big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T\cdot\mathbf{e}_\chi \qquad (3.1)\end{aligned}$$

where $\|\big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T\cdot\mathbf{e}_\chi\| \le |\big(\mathbf{G}^{-1}(\mathbf{c}_I)\big)^T|\cdot|\mathbf{e}_\chi| \le \hat{n}_I\cdot B$ by Lemma 2.6. □

Lemma 3.2 (Security). *Set $sk_I \in \mathbb{Z}^{n_I}$ be any vector. Suppose we generate $sk_O \leftarrow \mathsf{SecretKeyGen}(\mathsf{params})$ and $\mathbf{P}_{I:O} \leftarrow \mathsf{SwitchKeyGen}(sk_I, sk_O)$, then there is no efficient adversary that can tell the difference between the distribution $\mathbf{P}_{I:O}$ and uniform distribution over $\mathbb{Z}_q^{\hat{n}_I\times n_O}$ under decisional $\mathsf{LWE}_{n,m,\chi,q}$ assumption.*

3.2 Homomorphic Computation

Actually, we just use an encoding hash function $\mathcal{H} : \mathbb{Z}_q^{m\times n}$ to map identities to matrices, i.e., represent the identities as matrices in $\mathbb{Z}_q^{m\times n}$ for some n and each entry is over $\mathbb{Z}_q^m$. In the following, we denote $\{0,1\}$ as the message space and $\mathcal{ID} = \{0,1\}^\kappa$ as the identity space for $\lambda \in \mathbb{N}$. In our IBPRE scheme, there exists an efficiently computable injective map S that maps an identity $id \in \{0,1\}^\kappa$ to a subset $S(id) = (y_1, \cdots, y_d)$ of $[1, \xi]^d$, where $\xi = \lceil \kappa^{1/d} \rceil$ and $y_\iota \in [\xi]$ are entries of the subset $S(id)$ for $\iota \in [d]$.

Definition 3.3 (From [7]). *We denote the recursive function $\mathbf{B}_j^*$ (i.e., $\mathsf{Eval}_d : (\mathbb{Z}_q^{n\times m})^d \to \mathbb{Z}_q^{n\times m}$) by $\mathbf{B}_j^* = \mathsf{Eval}_d(\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_d) := \mathbf{B}_1\cdot\mathbf{G}^{-1}(\mathbf{B}_{j-1}^*)$ for $j \ge 1$, which takes a set of matrices $\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_d \in \mathbb{Z}_q^{n\times m}$ as input and outputs matrix $\mathbf{B}_j^*$ in $\mathbb{Z}_q^{n\times m}$. Note that*

$$\begin{aligned}\mathbf{B}_j^* &= \mathsf{Eval}_d(\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_d) \\ &= \mathbf{B}_1\cdot\mathbf{G}^{-1}\Big(\mathsf{Eval}_{d-1}(\mathbf{B}_2, \cdots, \mathbf{B}_d)\Big)\end{aligned}$$

Definition 3.4 (From [7]). *We denote the recursive function $\mathbf{R}_j^*$ (i.e., $\mathsf{TrapEval}_d : (\mathbb{Z}_q^{n\times m})^d \to \mathbb{Z}_q^{n\times m}$) by*

$$\begin{aligned}\mathbf{R}_j^* &= \mathsf{TrapEval}_d(\{\mathbf{R}_j\}, \{y_j\})_{j\in[d]} \\ &= \mathbf{R}_1\mathbf{G}^{-1}\Big(\mathbf{R}_{j-1}^*\Big) + y_1\Big(\mathbf{R}_{j-1}^*\Big)\end{aligned}$$

which takes a set of matrices $\mathbf{R}_1, \cdots, \mathbf{R}_d \in \mathbb{Z}_q^{n\times m}$ as input and outputs matrix $\mathbf{R}_j^*$ in $\mathbb{Z}_q^{n\times m}$.

Remark 3.5. *Observe that $\mathbf{R}_j^* := \mathbf{R}_1$ for $d = 1$; if $d \ge 2$, then there exists*

$$\begin{aligned}\mathbf{R}_j^* &= \mathsf{TrapEval}_d(\{\mathbf{R}_j\}, \{y_j\})_{j\in[d]} \\ &= \mathbf{R}_1\cdot\mathbf{G}^{-1}\Big(\mathsf{TrapEval}_{d-1}(\{\mathbf{R}_j\}, \{y_j\})\Big) \\ &\quad + y_1\Big(\mathsf{TrapEval}_{d-1}(\{\mathbf{R}_j\}, \{y_j\})\Big).\end{aligned}$$

In this setting, for future convenience, we re-write

$$\mathbf{R}_j^* := \mathbf{R}_j\mathbf{G}^{-1}\Big(\mathbf{A}\mathbf{R}_{j-1}^* + (\Pi_{i=1}^{j-1}y_j)\mathbf{G} + y_j\mathbf{R}_{j-1}^*\Big).$$

Lemma 3.6 ([7]). *If there exists $\mathbf{B}_i = \mathbf{A}\cdot\mathbf{R}_i + y_i\cdot\mathbf{G}$ for $\mathbf{A}$, $\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{R}_i \in \mathbb{Z}_q^{m\times m}$ with $i \in [d]$, then it holds that: $\mathbf{B}^* := \mathsf{Eval}_d(\mathbf{B}_1, \cdots, \mathbf{B}_d) = \mathbf{A}\mathbf{R}_F + \mathbf{F}(y)\mathbf{G}$ where $\mathbf{R}_F$ is generated by $\mathsf{TrapEval}_d(\mathbf{R}_1, \cdots, \mathbf{R}_d, y_1, \cdots, y_d)$ and satisfies that $\|\mathbf{R}_F\| \le m\cdot d\delta^{d-1}$.*

Proof: Due to $\mathbf{B}_i = \mathbf{A}\cdot\mathbf{R}_i + y_j\cdot\mathbf{G}$ for $i \in [d]$. Hence,

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

$$\begin{aligned}\mathbf{B}^* &:= \mathsf{Eval}_d(\mathbf{B}_1, \cdots, \mathbf{B}_d)\\ &= \mathbf{B}_d \cdot \mathbf{G}^{-1}\Big(\mathsf{Eval}_{d-1}(\mathbf{B}_1, \cdots, \mathbf{B}_{d-1})\Big)\\ &= \mathbf{A}\Big(\mathbf{R}_d\mathbf{G}^{-1}\Big(\mathbf{A}\mathbf{R}^*_{d-1} + (\Pi_{j=1}^{d-1} y_j)\mathbf{G}\Big) + y_j\mathbf{R}^*_{d-1}\Big)\\ &\quad + (\Pi_{j=1}^{d} y_j)\mathbf{G} \;=\; \mathbf{A}\mathbf{R}^*_d + (\Pi_{j=1}^{d} y_j)\mathbf{G}\end{aligned}$$

Therefore, $\|\mathbf{R}^*\| \le \|\mathbf{R}_d \cdot \mathbf{G}^{-1}\Big(\mathbf{A}\cdot\mathbf{R}^*_{d-1} + (\Pi_{j=1}^{d-1} y_j)\mathbf{G}\Big)\|_\infty + |x_d| \cdot \|\mathbf{R}^*_{d-1}\|_\infty \le m \cdot \|\mathbf{R}_d\|_\infty + 1 \cdot \|\mathbf{R}_{d-1}\|_\infty = m \cdot \delta + m(d-1) \cdot \delta = md\delta$ since $\mathbf{G}^{-1}(\cdot) \in \{0,1\}^{m\times m}$. □

Corollary 3.7. *For an injective map $S : \{0,1\}^\kappa \to 2^{[d]\times[\xi]}$ that maps an identity id to a subset of the set $[d]\times[\xi]$. Hence, for $2 \le i \in [d],\ 2 \le j \in [\xi]$, we have*

$$\begin{aligned}\mathcal{H}(id) &= \mathbf{B}_0 + \sum_{(i,j)\in S(id)} \mathbf{B}_{i,j_i} \cdot \mathbf{G}^{-1}(\mathbf{B}^*_{i-1,j-1_{i-1}})\\ &= \mathbf{A}\cdot\mathbf{R}_{id} + F_{\mathbf{y}}(id)\cdot\mathbf{G} \in \mathbb{Z}_q^{m\times n}, \qquad (3.2)\end{aligned}$$

where $\mathbf{R}_{id} = \mathbf{R}_{i,j_i}\mathbf{G}^{-1}\Big(\mathbf{R}^*_{i-1,j-1_{i-1}}\Big) + y_i\Big(\mathbf{R}^*_{i-1,j-1_{i-1}}\Big)$ *and* $F_{\mathbf{y}}(id) = y_0 + \sum_{(i,j)\in S(id)} y_{1,j_1}\cdots y_{d,j_d}$.

3.3 Single-Hop Homomorphic IBPRE Scheme

In this subsection, in order to obtain single-hop homomorphic IBPRE scheme, informally, we first assemble the IBE scheme of Yamada [7] into PRE scheme [11], then, armed with our GPV-style key-switching mechanism, we construct IBPRE scheme by fully key homomorphic computation [32]. However, we easily find that the above IBPRE scheme only supports addition homomorphic operation and limited times multiplication homomorphic operation since the expansion of noise. Our ultimate goal is to obtain a multi-hop homomorphic IBPRE scheme. Hence, we need to tweak the encryption algorithm and make it support homomorphic (i.e., addition and multiplication) operation. Inspired by the work of Li et al. [49], which proposed a multi-bit FHE scheme via dual Regev scheme. Thus we can work along Li et al.'s technical line, and construct a homomorphic IBPRE scheme. Below, we give a detailed description of homomorphic IBPRE construction.

- $(mpk, msk) \leftarrow \mathsf{IBPRE.Setup}(1^\lambda)$:
  1) Samples random matrices $\mathbf{B}_0 \leftarrow \mathbb{Z}_q^{m\times n}$ and $\mathbf{B}_{\iota,k} \leftarrow \mathbb{Z}_q^{m\times n}$ for $(\iota,k) \in [d,\xi]$. Then, draw a vector $\mathbf{u}$ from distribution $\mathbb{Z}_q^{m\times 1}$;
  2) Invokes the sub-procedure $(\mathbf{A} \in \mathbb{Z}_q^{m\times n}, \mathbf{T}_\mathbf{A}) \in \mathbb{Z}^{m\times m} \leftarrow \mathsf{TrapGen}(m,n,q)$ s.t. the trapdoor $\mathbf{T}_\mathbf{A}$ satisfies the GM norm $\|\mathbf{T}_\mathbf{A}\|_{GS} = O\sqrt{n\log q}$;
  3) Outputs $mpk = (\mathbf{A}, \mathbf{B}_0, \{\mathbf{B}_{\iota,k}\}_{(\iota,k)\in[d,\xi]}, \mathbf{u})$ and $msk = \mathbf{T}_\mathbf{A}$.

  Remark 3.8. *Consider the master public key contains a uniform random parity-check matrix $\mathbf{A}$, random matrix $\mathbf{B}_0$, and a sequence of $\mathbf{B}_{\iota,k}$ for $(\iota,k) \in [d,\xi]$. The master secret key is a trapdoor for matrix $\mathbf{A}$.*
- $sk_{id} \leftarrow \mathsf{IBPRE.KeyGen}(mpk, msk, id)$:
  1) Utilizing a deterministic hash function $\mathcal{H} : id \to \mathbb{Z}_q^{m\times n}$ to map each user's identity id to $\mathcal{H}(id)$, then it holds that
  $$\mathcal{H}(id) = \mathbf{B}_0 + \sum_{(i,j)\in S(id)} \mathbf{B}_{i,j_i} \cdot \mathbf{G}^{-1}(\mathbf{B}^*_{i-1,j-1_{i-1}})$$
  2) Then we associate each identity id with the user-specific matrix $\mathbf{A}_{id} := [\mathbf{A}|\mathcal{H}(id)] \in \mathbb{Z}_q^{m\times 2n}$;
  3) Picks a vector $\mathbf{e} \in \mathbb{Z}_q^{2n\times 1}$, then such that $\mathbf{A}_{id}\cdot\mathbf{e} = \mathbf{u} \pmod q \in \mathbb{Z}_q^{m\times 1}$, by the algorithm $\mathsf{SampleLeft}(\mathbf{A}, \mathcal{H}(id), \mathbf{u}, \mathbf{T}_\mathbf{A}, \sigma) \to \mathbf{e}$;
  4) Sets and outputs the private key $sk_{id} := \mathbf{s} = [1, -\mathbf{e}^T] \in \mathbb{Z}_q^{1\times(2n+1)}$. Notably, the key observation is that $(\mathbf{u}, \mathbf{A}_{id}) \cdot [1, -\mathbf{e}^T]^T = 0 \pmod q$.
- $\mathbf{c} \leftarrow \mathsf{IBPRE.Enc}(mpk, id, \mu \in \{0,1\})$: Homomorphic encryption algorithm with short ciphertexts follows by Brakerski et al. [45].
  1) First samples a uniform random vector $\mathbf{r}$ from $\{0,1\}^{m\times 1}$, and 1-th standard basis vector $\mathbf{t}_1$;
  2) Samples a random vector $\mathbf{x}$ from Gaussian distribution $\chi^{(2n+1)\times 1}$
  3) Computes and outputs the encryption $\mathbf{c} := (\mathbf{u} \mid \mathbf{A}_{id})^T \cdot \mathbf{r} + \mu \cdot (2^{\ell_q - 1}\mathbf{t}_1) + \mathbf{x} \pmod q \in \mathbb{Z}_q^{(2n+1)\times 1}$.

  Remark 3.9. *We remark that, in this paper, we tweak the encryption algorithm and make it support homomorphic operation. Moreover, if we fully adopt the encryption algorithm of Li et al. [49], the dimension of ciphertext will expand, namely that the dimension of ciphertext will expand from $\mathbf{c} \in \mathbb{Z}_q^{(2n+1)\times 1}$ to $\mathbf{C} := \mu\mathbf{G} + (\mathbf{u}, \mathbf{A}_{id})^T\mathbf{R} + \mathbf{X} \pmod q \in \mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$ for the gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$ and error matrix $\mathbf{X} \in \chi^{(2n+1)\times(2n+1)\ell_q}$. To address this issue, we adopt the technique of Brakerski et al. [45] which constructs a multi-key fully homomorphic encryption with short ciphertexts. Actually, the only difference is that the encryption algorithm [45] is achieved by a fragment of $\mathbf{G}$ rather than the whole $\mathbf{G}$.*
- $rk^{(i\to j)} \leftarrow \mathsf{IBPRE.ReKeyGen}(mpk, id^{(j)}, id^{(i)}, sk_{id^{(i)}})$: In *single-hop re-encryption* setting, without loss of generality, we first assume the re-encryption from player i to player $j := i+1$. Hence, at input side, i.e., player i with the secret key $sk_{id^{(i)}} := \mathbf{s}_i = (1, -\mathbf{e}_i^T) \in \mathbb{Z}_q^{1\times(2n+1)}$ which corresponds to the user-specific matrix $\mathbf{A}_{id^{(i)}}$. Similarly, we can easily obtain the secret key $sk_{id^{(j)}} := \mathbf{s}_j = (1, -\mathbf{e}_j^T)$ of player j at output side, and the user-specific matrix $\mathbf{A}_{id^{(j)}}$ of player j. Thus, it holds that,
  1) Samples a random matrix $\mathbf{R}^{(j)} \leftarrow \{-1,0,1\}^{m\times m}$ and computes
  $$\mathbf{N}^{(j)} := (\mathbf{u}|\mathbf{A}_{id^{(j)}})^T \cdot \mathbf{R}^{(j)} \in \mathbb{Z}_q^{(2n+1)\times m};$$
  2) Samples a random matrix $\mathbf{R}^{(i\to j)} \leftarrow \{-1,0,1\}^{m\times(2n+1)\ell_q}$ and a random vector $\mathbf{z}_j \leftarrow \{-1,0,1\}^{1\times(2n+1)\ell_q}$, then computes and outputs $\mathbf{M}^{(i\to j)} \in \mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$, where
  $$\mathbf{M}^{(i\to j)} \leftarrow \left[\frac{\mathbf{u}^T\cdot\mathbf{R}^{(i\to j)} + \big([1, -\mathbf{e}_i^T]\cdot\mathbf{G}\big) + \mathbf{z}_j}{\mathbf{A}_{id^{(j)}}^T\cdot\mathbf{R}^{(i\to j)}}\right];$$
  3) Outputs re-encryption key $rk^{(i\to j)} := \mathbf{K}^{(i\to j)} = (\mathbf{M}^{(i\to j)}, \mathbf{N}^{(j)})$.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

- $\mathbf{c}_{id^{(j)}} \leftarrow \mathsf{ReEnc}(mpk, rk^{(i\to j)}, \mathbf{c}_{id^{(i)}})$: For readability, we rewrite $\mathbf{c}_{id^{(i)}}$ (and $\mathbf{c}_{id^{(j)}}$) as $\mathbf{c}_i$ (and $\mathbf{c}_j$). In order to achieve the re-encryption, we need to embed the ciphertext and re-encryption key into NAND gate circuit, i.e., the re-encryption ciphertext is $\mathbf{K}\cdot\mathbf{G}^{-1}(\mathbf{c})$ for a re-encryption key $\mathbf{K}$ and the ciphertext $\mathbf{c}$ at input side.
  1) Samples random vectors $\hat{\mathbf{r}} \in \{0,1\}^{m\times 1}$, and $\mathbf{y} := (y_0 \leftarrow \chi, \mathbf{y}_1 \leftarrow \chi^{2n\times 1}) \leftarrow \chi^{(2n+1)\times 1}$;
  2) Computes and outputs

$$\mathbf{c}_j := \mathbf{M}^{(i\to j)}\cdot\mathbf{G}^{-1}(\mathbf{c}_i) + \mathbf{N}^{(j)}\cdot\hat{\mathbf{r}} + \mathbf{y} \in \mathbb{Z}_q^{(2n+1)\times 1},$$

  where $\mathbf{G}^{-1}(\mathbf{c}_i) \in \mathbb{Z}_q^{(2n+1)\ell_q\times 1}$.
- $\mathsf{Dec}(\mathsf{params}, sk_L, \mathbf{c}_L)$: There also exist two cases.
  - Decryption at input side, computes and outputs the results of $\langle \mathbf{c}, \mathbf{s}\rangle \pmod q$,

$$\langle \mathbf{c}, \mathbf{s}\rangle := \mu\cdot 2^{\ell_q-1} + \mathbf{sx} \pmod q,$$

    If $\mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}) := \|\mathbf{sx}\| < q/8$, then sets $\mu = 1$ and otherwise sets $\mu = 0$. Outputs μ.
  - Decryption at output side, i.e., decrypts the re-encryption ciphertexts. In this setting, the re-encryption ciphertext is $\mathbf{c}_j := \mathbf{K}^{(i\to j)}\cdot\mathbf{G}^{-1}(\mathbf{c}_i)$ for $j = i+1$, hence the decrypter computes and outputs the results of $\langle \mathbf{c}_j, \mathbf{s}_j\rangle = \mu 2^{\ell_q-1} + \mathsf{noise}_{(\mathbf{s}_j,\mu)}(\mathbf{c}_j) \pmod q$, if $\mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}) := \|\mathbf{error}\| \le q/8$ for $\mathbf{error} := x_1 - \mathbf{x}_2^T\cdot\mathbf{e}_i + y_1 - \mathbf{y}_2^T\cdot\mathbf{e}_j + \mathbf{z}_j\mathbf{G}^{-1}(\mathbf{c}_i)$, then outputs $\mu = 1$ and otherwise $\mu = 0$.
- $\mathsf{IBPRE.Eval}(\mathsf{params}, id, mpk, \mathbf{c}_1, \cdots, \mathbf{c}_n)$: There exist two types of homomorphic operation. For ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ under the same master public key mpk and identity id, we have that,
  - **Homomorphic Addition:** $\mathsf{Add}(mpk, id, \mathbf{c}_1, \mathbf{c}_2)$: Computes and outputs $\mathbf{c}_1 + \mathbf{c}_2 = (\mu_1+\mu_2)\cdot(2^{\ell_q-1}\mathbf{t}_1) + \mathbf{A}^T(\mathbf{r}_1+\mathbf{r}_2) + (\mathbf{x}_1+\mathbf{x}_2) \pmod q$;
  - **Homomorphic Multiplication:** $\mathsf{Mult}(mpk, id, \mathbf{C}_1, \mathbf{c}_2)$: Computes and outputs

$$\begin{aligned}&\mathbf{C}_1\cdot\mathbf{G}^{-1}(\mathbf{c}_2)\\ &= (\mu_1\mathbf{G} + (\mathbf{u}, \mathbf{A}_{id})^T\mathbf{R}_1 + \mathbf{X}_1)\cdot\mathbf{G}^{-1}(\mathbf{c}_2)\\ &= \mu_1\mu_2\cdot(2^{\ell_q-1}\mathbf{t}_1) + \mu_1(\mathbf{u}, \mathbf{A}_{id})^T\mathbf{r}_2 + \mu_1\mathbf{x}_2\\ &\quad + (\mathbf{u}, \mathbf{A}_{id})^T\mathbf{R}_1\cdot\mathbf{G}^{-1}(\mathbf{c}_2) + \mathbf{X}_1\mathbf{G}^{-1}(\mathbf{c}_2) \pmod q\end{aligned}$$

    where $\mathbf{C}_1 = \mu_1\mathbf{G} + (\mathbf{u}, \mathbf{A}_{id^{(1)}})^T\mathbf{R}_1 + \mathbf{X}_1 \pmod q$ for a random matrix $\mathbf{R}_1 \in \{0,1\}^{m\times(2n+1)\ell_q}$ and an error matrix $\mathbf{X}_1 \in \chi^{(2n+1)\times(2n+1)\ell_q}$.

3.4 Correctness

Below, we analyze the bound of homomorphic noise.

Definition 3.10 ([46], [45]). *If the short ciphertext of dual GSW scheme is* $\mathbf{c} = \mu 2^{\ell_q-1}\mathbf{t}_1 + (\mathbf{u}, \mathbf{A}_{id})^T\cdot\mathbf{r} + \mathbf{x} \in \mathbb{Z}_q^{(2n+1)\times 1}$, *along with the secret key* $\mathbf{s} = (1, -\mathbf{e}^T) \in \mathbb{Z}_q^{1\times(2n+1)}$, *random noise vector* $\mathbf{x} \in \mathbb{Z}_q^{(2n+1)\times 1}$, *and* $2^{\ell_q-1}\mathbf{t}_1 \in \mathbb{Z}_q^{(2n+1)\times 1}$, *then the noise of* $\mathbf{c}$ *is the infinity norm of the noise vector:* $\mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}) = \|\mathbf{c} - \mu 2^{\ell_q-1}\mathbf{t}_1\|_\infty$, *i.e.,* $\mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}) = \|\mathbf{s}\cdot(\mathbf{u}, \mathbf{A}_{id})^T\mathbf{r} + \mathbf{s}\cdot\mathbf{x}\| = \|\mathbf{0} + \mathbf{s}\cdot\mathbf{x}\| \le \|x_1 + \mathbf{x}_2^T\cdot\mathbf{e}\| \le \frac{q}{8}$.

Lemma 3.11. *For the ciphertexts* $\mathbf{c} = \mu 2^{\ell_q-1}\mathbf{t}_1 + (\mathbf{u}, \mathbf{A}_{id})^T\cdot\mathbf{r} + \mathbf{x} \in \mathbb{Z}_q^{(2n+1)\times 1}$, *along with the secret key* $\mathbf{s} = (1, -\mathbf{e}^T)^T \in \mathbb{Z}_q^{2(n+1)\times 1}$ *and* $2^{\ell_q-1}\mathbf{t}_1 \in \mathbb{Z}_q^{(2n+1)\times 1}$, *then the noise in negation, addition and multiplication is bounded as follows:*

- ***Addition:*** *For all messages* $\mu_1, \mu_2 \in \{0,1\}$, *it holds that*

$$\mathsf{noise}_{(\mathbf{s},\mu_1+\mu_2)}(\mathbf{c}_1+\mathbf{c}_2) \le \mathsf{noise}_{(\mathbf{s},\mu_1)}(\mathbf{c}_1) + \mathsf{noise}_{(\mathbf{s},\mu_2)}(\mathbf{c}_2);$$

- ***Multiplication:*** *For all messages* $\mu_1, \mu_2 \in \{0,1\}$, *it holds that*

$$\begin{aligned}\mathsf{noise}_{(\mathbf{s},\mu_1\mu_2)}(\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{c}_2)) &\le \mu_1\cdot\mathsf{noise}_{(\mathbf{s},\mu_2)}(\mathbf{c}_2)\\ &\quad + (2n+1)\ell_q\cdot\mathsf{noise}_{(\mathbf{s},\mu_1)}(\mathbf{C}_1)\end{aligned}$$

  for an efficiently computable function $\mathbf{G}^{-1}$: $\mathbb{Z}_q^{(2n+1)\ell_q} \to \mathbb{Z}_q^{2n+1}$. *i.e.,*

$$\begin{aligned}\mathsf{noise}_{(\mathbf{s},\mu_1\mu_2)}(\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{c}_2)) &\le \|\mu_1\cdot(\mathbf{s}^T\mathbf{x}_2)\\ &\quad + (\mathbf{s}^T\mathbf{X}_1)\cdot\mathbf{G}^{-1}(\mathbf{c}_2)\|.\end{aligned}$$

  We note that, $\mathbf{c}_2 = \mu_2 2^{\ell_q-1}\mathbf{t}_1 + (\mathbf{u}, \mathbf{A}_{id})^T\mathbf{r}_2 + \mathbf{x}_2$ *over* $\mathbb{Z}_q^{1\times(2n+1)}$ *and* $\mathbf{C}_1 = \mu_1\mathbf{G} + (\mathbf{u}, \mathbf{A}_{id})^T\mathbf{R}_1 + \mathbf{X}_1$ *over* $\mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$.
- ***Negation:*** *For all messages* $\mu \in \{0,1\}$, *it holds that*

$$\mathsf{noise}_{(\mathbf{s},1-\mu)}(2^{\ell_q-1}\mathbf{t}_1 - \mathbf{c}) = \mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}).$$

Lemma 3.12. *We say the homomorphic* IBPRE *scheme is correct at input side if the noise is bounded by* $E \le \frac{q}{8}$.

Proof: Consider the ciphertexts $\mathbf{c}_{id^{(i)}}$ and secret key $sk_{id^{(i)}}$ at input side, we hold that

$$\begin{aligned}&\langle \mathbf{c}_{id^{(i)}}, sk_{id^{(i)}}\rangle\\ =\ &\left((\mathbf{u}, \mathbf{A}_{id^{(i)}})^T\cdot\mathbf{r} + \mu 2^{\ell_q-1}\mathbf{t}_1 + \mathbf{x}\right)^T\cdot\begin{pmatrix}1\\ -\mathbf{e}_i\end{pmatrix}\\ \le\ &\mu 2^{\ell_q-1} + \mathsf{noise}_{(\mathbf{s}_i,\mu)}(\mathbf{c}_i) \pmod q;\end{aligned}$$

If and only if $\mathsf{noise}_{(\mathbf{s},\mu)}(\mathbf{c}) := \|x_1 - \mathbf{x}_2^T\cdot\mathbf{e}_i\| \le |x_1| + |\mathbf{x}_2^T\cdot\mathbf{e}_i| \le E \le \frac{q}{8}$ via Lemma 2.8, we can correctly decrypt. □

Definition 3.13. *If the ciphertexts of output side* $\mathbf{c}_j := \mathbf{M}^{(i\to j)}\cdot\mathbf{G}^{-1}(\mathbf{c}_i) + \mathbf{N}^{(j)}\cdot\hat{\mathbf{r}} + \mathbf{y} \in \mathbb{Z}_q^{(2n+1)\times 1}$, *along with the secret key* $\mathbf{s}_j = (1, -\mathbf{e}_j^T) \in \mathbb{Z}_q^{1\times(2n+1)}$ *at the output side, then the noise of* $\mathbf{c}_j$ *is the infinity norm of the noise vector:* $\mathsf{noise}_{(\mathbf{s}_j,\mu)}(\mathbf{c}_j) = \|\mathsf{noise}_{(\mathbf{s}_i,\mu)}(\mathbf{c}_i) + \mathbf{y}^T\cdot\mathbf{s}_j\| = \|\mathsf{noise}_{(\mathbf{s}_i,\mu)}(\mathbf{c}_i) + y_1 - \mathbf{y}_2^T\cdot\mathbf{e}_j\| \le \mathsf{noise}_{(\mathbf{s}_i,\mu)}(\mathbf{c}_i) + \|y_1 - \mathbf{y}_2^T\cdot\mathbf{e}_j\| \le \mathsf{noise}_{(\mathbf{s}_i,\mu)}(\mathbf{c}_i)$.

Lemma 3.14. *We say the* IBPRE *scheme is correct at output side if the noise is bounded by* $E \le \frac{q}{8}$.

Proof: Consider the ciphertexts $\mathbf{c}_j$ and secret key $sk_{id^{(j)}}$ at output side, we hold that

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

$$
\begin{aligned}
\langle \mathbf{c}_j, \mathbf{s}_j \rangle &= \Big(\mathbf{M}^{(i\to j)} \cdot \mathbf{G}^{-1}(\mathbf{c}_i) + \mathbf{N}^{(j)} \cdot \hat{\mathbf{r}} + \mathbf{y}\Big)^T \cdot \mathbf{s}_j \\
&= \Bigg(\left[\frac{\mathbf{u}_j^T \mathbf{R}^{(i\to j)} + \big([1, -\mathbf{e}_i^T]\mathbf{G}\big) + \mathbf{z}_j}{(\mathbf{A}_{id^{(j)}})^T \cdot \mathbf{R}^{(i\to j)}} \right] \cdot \mathbf{G}^{-1}(\mathbf{c}_i) \\
&\quad + \mathbf{P}_j^T \cdot \mathbf{R}^{(j)} \cdot \hat{\mathbf{r}} + \mathbf{y} \Big)^T \cdot \left(\frac{1}{-\mathbf{e}_j} \right) \\
&= \big([1, -\mathbf{e}_i^T]\mathbf{G}\mathbf{G}^{-1}(\mathbf{c}_i)\big) + \underline{\big(\mathbf{R}^{(i\to j)}\mathbf{G}^{-1}(\mathbf{c}_i)\big)^T \mathbf{u}_j} \\
&\quad - \underline{\big(\mathbf{R}^{(i\to j)}\mathbf{G}^{-1}(\mathbf{c}_i)\big)^T (\mathbf{A}_{id^{(j)}}\mathbf{e}_j)} + \mathbf{z}_j \cdot \mathbf{G}^{-1}(\mathbf{c}_i) \\
&\quad + (y_1 - \mathbf{y}_2^T \mathbf{e}_j) \\
&= \mu \cdot 2^{\ell_q - 1} + x_1 - \mathbf{x}_2^T \mathbf{e}_i + \mathbf{z}_j \mathbf{G}^{-1}(\mathbf{c}_i) + y_1 - \mathbf{y}_2^T \mathbf{e}_j \\
&\leq \mu \cdot 2^{\ell_q - 1} + \mathsf{noise}_{(\mathbf{s}_j, \mu)}(\mathbf{c}_j) \pmod q
\end{aligned}
$$

If and only if $\mathsf{noise}_{(\mathbf{s}_j,\mu)}(\mathbf{c}_j) = \|x_1 - \mathbf{x}_2^T \cdot \mathbf{e}_i + y_1 - \mathbf{y}_2^T \cdot \mathbf{e}_j + \mathbf{z}_j \mathbf{G}^{-1}(\mathbf{c}_i)\| \leq \|x_1 - \mathbf{x}_2^T \mathbf{e}_i\| + \|y_1 - \mathbf{y}_2^T \mathbf{e}_j\| + \|\mathbf{z}_j \mathbf{G}^{-1}(\mathbf{c}_i)\| \leq \big((2n+1)\ell_q\big)^2 B \leq \frac{q}{4}$, outputs $\mu = 1$. □

3.5 Security Analysis

In order to satisfy the work of security proof, we need the following theorem to address it.

Theorem 3.15. *The above* IBPRE *scheme is* IND-ID-*CPA (semantic) secure assuming the* $\mathsf{LWE}_{n,m,\chi,q}$ *is hard.*

Proof: We show the security via the following hybrids games. Let Game_0 be the interaction between $\mathcal{A}$ and $\mathcal{C}$ in above definition 2.20.

- Game_0 : This is identical to the IND-ID-CPA with the real security game, where the adversary $\mathcal{A}$ gets properly the master public key mpk generated by $\mathsf{Setup}(\cdot)$, uncorrupted secret key sk_{id} generated by $\mathsf{KeyGen}(\cdot)$, re-encryption key $rk^{(i\to j)}$ generated by $\mathsf{ReKeyGen}(\cdot)$, and an encryption of 0 or 1 computed by $\mathsf{Enc}(\cdot)$. Moreover, in the challenge phase, the challenge ciphertext is set as $\mathbf{c}^* \leftarrow \mathbb{Z}_q^{(2n+1)\times 1}$ if $b = 1$. Hence, at the end of the game, $\mathcal{A}$ outputs a guess bit b' for b. Finally, the challenger $\mathcal{C}$ sets $b' = b$. Hence, by the definition, the advantage of adversary $\mathcal{A}$ is $\mathbf{Adv}[\mathcal{A}] \triangleq |\Pr[b' = b]|$.
- Game_1 : This game is identical to Game_0 in everything except the generation of the challenge identity in key extraction queries. Below we sketch this game and refer the reader to find more details from [7]. The challenger $\mathcal{C}$ first picks $\mathbf{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[d]\times[\xi]})$ as $y_0, \{y_{i,j}\}_{(i,j)\in[d]\times[\xi]} \in \mathbb{Z}_q$ for $(i,j) \in [d] \times [\xi]$. For future convenience, we define a function $F_{\mathbf{y}}(id)$ at Corollary 3.7 which satisfies the following condition
$$F_{\mathbf{y}}(id^*) = 0 \bigwedge F_{\mathbf{y}}(id_1) \neq 0 \bigwedge \cdots \bigwedge F_{\mathbf{y}}(id_Q) \neq 0,$$
for the challenge identity id^*. The adversary $\mathcal{A}$ has made key extraction queries identity $id_1, \cdots, id_Q$, where $Q < q$. As we show in Lemma 3.16, it follows that there exists an adversary $\mathcal{A}_1$ with advantage $\mathbf{Adv}[\mathcal{A}] \leq \mathbf{Adv}[\mathcal{A}_1] + \delta$.
- Game_2 : This game is identical to Game_1 in everything except the generation of mpk. Considering the distributions of mpk
$$\Big(\mathbf{A}, \mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}, \{\mathbf{A}\mathbf{R}_{i,j} + y_{i,j}\mathbf{G}\}_{(i,j)\in[d]\times[\xi]}\Big)$$
and a uniform distribution
$$\Big(\mathbf{A}, \mathbf{B}_0, \{\mathbf{B}_{i,j}\}_{(i,j)\in[d]\times[\xi]}\Big)$$
are statistical indistinguishable via leftover hash lemma, where $\mathbf{B}_0, \{\mathbf{B}_{i,j}\}_{(i,j)\in[d]\times[\xi]} \leftarrow \mathbb{Z}_q^{n\times m}$. Now, we have the advantage $\mathbf{Adv}[\mathcal{A}_1] \leq \mathbf{Adv}[\mathcal{A}_2] + \delta_1$.
- Game_3 : This game is identical to Game_2 in everything except the user-specific matrix $\mathbf{A}_{id}$. In more detail: for the user-specific matrix $\mathbf{A}_{id} := [\mathbf{A}, \mathcal{H}(id)] \in \mathbb{Z}_q^{m\times 2n}$, where $\mathcal{H}(id) = \mathbf{A} \cdot \mathbf{R}_{id} + F_{\mathbf{y}}(id) \cdot \mathbf{G}$. If $F_{\mathbf{y}}(id) = 0$ and aborts, otherwise, the challenger $\mathcal{C}$ samples $\widehat{\mathbf{A}}_{id} \leftarrow \mathbb{Z}_q^{m\times 2n}$, and $\mathbf{A}_{id} \approx_s \widehat{\mathbf{A}}_{id}$ by leftover hash lemma. Namely, it follows that there exists $\mathbf{Adv}[\mathcal{A}_2] \leq \mathbf{Adv}[\mathcal{A}_3] + \delta_2$.
- Game_4 : This game is identical to Game_3 in everything except: **1).** the way of the key extraction queries are answered and **2).** the way of the matrix $\mathbf{A}$ is sampled without trapdoor, i.e., $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$. In this game, the adversary $\mathcal{A}$ first makes a key extraction query $\mathsf{KeyGen}(\cdot)$ for an identity id, then, the challenger $\mathcal{C}$ checks whether $F_{\mathbf{y}}(id) = 0$ and aborts, otherwise, $\mathcal{C}$ computes $\mathbf{R}_{id}$ via the function $\mathsf{TrapEval}$ as the definition of $\mathbf{R}_{id}$. Hence, we have that $[\mathbf{A}|\mathcal{H}(id)] \cdot \mathbf{e} = \mathbf{u}$ for $\mathcal{H}(id) = \mathbf{A} \cdot \mathbf{R}_{id} + F_{\mathbf{y}}(id) \cdot \mathbf{G}$., i.e., the challenger generates $\mathbf{e} \leftarrow \mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{id}, F_{\mathbf{y}}(id), \mathbf{u}, \mathbf{T}_{\mathbf{G}}, \sigma)$, as the secret key, then sends $\mathbf{e}$ to $\mathcal{A}$. [1]
In this argument, we note that identity id of our IBPRE is generated by $\mathsf{SampleLeft}$ with the trapdoor basis $\mathbf{T}_{\mathbf{A}}$ of $\Lambda_q^{\perp}(\mathbf{A})$. Therefore, we say that the output distribution of $\mathsf{SampleRight}$ is indistinguishable from the sample over $D^{\mathbf{u}}_{(\mathbf{A}|\mathcal{H}(id)),\sigma}$ via $\mathbf{R}_{id}$ (Corollary 3.7) and the choice of σ [2] by Lemma 2.11. Namely that the computational distance is δ_3, where δ_3 is negligible in λ. Hence, we have the advantage $\mathbf{Adv}[\mathcal{A}_3] \leq \mathbf{Adv}[\mathcal{A}_4] + \delta_3$.
- Game_5 : This game is identical to Game_4 in everything except the method of generating the re-encryption key. More specially, the adversary can't check the correctness of re-encryption ciphertext. Considering the re-encryption key distribution $rk^{(i\to j)} = \{\mathbf{N}^{(j)}, \mathbf{M}^{(i\to j)}\}$, the challenger first change the way of the generation of $\mathbf{N}^{(j)}$ by sampling from the distribution $\mathbb{Z}_q^{(2n+1)\times m}$ rather than by computing $\mathbf{A}_{id^{(j)}}^T \cdot \mathbf{R}^{(j)}$; secondly, the challenger change the way of the generation of $\mathbf{M}^{(i\to j)}$ by sampling from the distribution $\mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$ rather than by computing; lastly, the challenger sends $rk^{(i\to j)}$ to adversary. Since the vector $\mathbf{u}$ is fixed and the user-specific matrix $\mathbf{A}_{id^{(i)}} := (\mathbf{A}|\mathcal{H}(id^{(i)})) \in \mathbb{Z}_q^{m\times 2n}$

1. Recall that the secret key generated via $\mathbf{e} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathcal{H}(id), \mathbf{u}, \mathbf{T}_{\mathbf{A}}, \sigma)$, i.e., the matrix $\mathbf{A}$ sampled with a trapdoor $\mathbf{T}_{\mathbf{A}}$ s.t. $\mathbf{A} \cdot \mathbf{e} = \mathbf{u}$.
2. Actually, $\mathsf{SampleLeft} \approx_c D^{\mathbf{u}}_{(\mathbf{A}|H(id)),\sigma}$

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

is replaced by unform distribution in $\widehat{\mathbf{A}}_{id} \in \mathbb{Z}_q^{m\times 2n}$, then the join distribution of $rk^{(i\to j)}$

$$\Big(\frac{\mathbf{u}^T}{\mathbf{A}_{id^{(j)}}^T}\Big)\cdot \mathbf{R}^{(j)}, \Big(\frac{\mathbf{u}^T\cdot \mathbf{R}^{(i\to j)}+[1,-\mathbf{e}_i^T]\cdot \mathbf{G}+\mathbf{z}_j}{\mathbf{A}_{id^{(j)}}^T\cdot \mathbf{R}^{(i\to j)}}\Big)$$

remains computationally indistinguishable from random distribution by the LWE assumption. Hence, we can say that the re-encryption key over $\mathbb{Z}_q^{(2n+1)\times\left((2n+1)\ell_q+m\right)}$ is formed by the identities from user i and j and secret key from user i. The ciphertext of re-encryption is distributed identically as in Game_4. Namely, it follows that there exists $\mathbf{Adv}[\mathcal{A}_4]\le \mathbf{Adv}[\mathcal{A}_5]+\delta_4$.

- Game_6 : This game is the same as Game_5 except the generation of ciphertext "at input side", i.e., in this game, the adversary submits the challenged identity id^* and message μ to Enc oracle, and then, the challenge ciphertext is generated by assuming the message $\mu=0$. Hence, the form of the challenge ciphertext is as follows:

  $$\hat{\mathbf{c}}=(\mathbf{u}|\widehat{\mathbf{A}}_{id})^T\mathbf{r}+\mathbf{x}+0\cdot 2^{\ell_q-1}\mathbf{t}_1 \pmod q \in \mathbb{Z}_q^{(2n+1)\times 1}$$

  where $\mathbf{u}$ is fixed and matrix $\widehat{\mathbf{A}}_{id}$ is sampled uniformly from $\mathbb{Z}_q^{m\times 2n}$. Hence, utilizing the LWE assumption, the LWE instance $(\mathbf{u}|\widehat{\mathbf{A}}_{id})^T\mathbf{r}+\mathbf{x}$ is indistinguishable from uniform distribution $\mathbf{c}'\in\mathbb{Z}_q^{(2n+1)\times 1}$. In this setting, we have $\mathbf{Adv}[\mathcal{A}_5]\le \mathbf{Adv}[\mathcal{A}_6]+\delta_5$.
- Game_7 : This game is the same as Game_6 except the generation of ciphertext "at output side", i.e., in this game, the output side ciphertext is generated by invoking the deterministic re-encryption algorithm ReEnc, i.e., computing

  $$\mathbf{c}_j := \mathbf{M}^{(i\to j)}\cdot \mathbf{G}^{-1}(\mathbf{c}_i')+\mathbf{N}^{(j)}\cdot \hat{\mathbf{r}}+\mathbf{y}\in\mathbb{Z}_q^{(2n+1)\times 1}.$$

  Thus, the adversary submits the challenged input side ciphertext $\mathbf{c}_i'$ to ReEnc oracle, and then, the challenger samples $\tilde{\mathbf{c}}$ from uniform distribution. Utilizing the leftover hash lemma, we can say $\tilde{\mathbf{c}}\approx_s \mathbf{c}_j'$. Hence, it follows that there exists $\mathbf{Adv}[\mathcal{A}_6]\le \mathbf{Adv}[\mathcal{A}_7]+\delta_6$.
  Most importantly, we note that, in Game_7, all the elements of the ciphertext, re-encryption key, secret key and user-specific matrix are uniformly random and independent of the message. Hence, we have $\mathbf{Adv}[\mathcal{A}_7]=1/2$.

In summary, putting them together, we have $\mathbf{Adv}[\mathcal{A}]<\mathbf{Adv}[\mathcal{A}_7]+\sum_i^6\delta_i = 1/2+\mathrm{negl}(\lambda)$, where δ_i $(i\in[6])$ is negligible in λ. That means, the adversary $\mathcal{A}$ can break the IND-ID-CPA-security of our IBPRE scheme with at most negligible advantage. This completes the proof. □

Moreover, in order to finish the security proof, it remains to show the following Lemma.

Lemma 3.16 (From [7]). *We first define* $\Gamma(\mathbb{ID})$ *as*

$$\Gamma(\mathbb{ID}) = \Pr_{\mathbf{y}}\Big[F_{\mathbf{y}}(id^*)=0 \bigwedge F_{\mathbf{y}}(id_1)\neq 0 \bigwedge \cdots \bigwedge F_{\mathbf{y}}(id_Q)\neq 0\Big]$$

for a sequence of identities $\mathbb{ID}=(id^*,id_1,\cdots,id_Q)\in\mathcal{ID}^{Q+1}$, *where the probability is taken over* $\mathbf{y}=(y_0,\{y_{i,j}\}_{(i,j)\in[d,\xi]})$.

If the above parameters are chosen as specific in Game_1, *and for any* $\mathbb{ID}=(id^*,id_1,\cdots,id_Q)$ *s.t.,* $id^*\neq id_i$ *for* $i\in[Q]$, *then there exist a upper bound and a lower bound for* $\Gamma(\mathbb{ID})$ *are all negligible in* λ , *i.e.,*

$$\frac{1}{\kappa+1}\cdot(\frac{1}{d\lambda^c})d\cdot(1-\frac{Q}{\lambda^c})\le\Gamma(\mathbb{ID})\le\frac{1}{\kappa+1}\cdot(\frac{1}{d\lambda^c})d$$

where c *is constant. Hence, for any* PPT *adversary* $\mathcal{A}_1$, *we have* $\mathbf{Adv}[\mathcal{A}_1]=\mathrm{negl}(\lambda)$ *which implies* $\mathbf{Adv}[\mathcal{A}]\le\mathbf{Adv}[\mathcal{A}_1]+\mathrm{negl}(\lambda)$ *(i.e.,* $\mathsf{Game}_0\approx_c\mathsf{Game}_1$*).*

The detailed proof of Lemma 3.16 will be found [7].

4 Multi-Hop Homomorphic IBPRE

In this section, we describe the construction of the multi-hop homomorphic IBPRE (mIBPRE). In order to transform the above single-hop homomorphic IBPRE into multi-hop homomorphic IBPRE, we still embed the ciphertext and re-encryption key into NAND gate circuit. Next, we transform the NAND gate into BP. In simple terms, for length L and width w, BP contains $(L+1)\cdot w$ nodes and each node (i,t) has two fan-out (or edges) $p_{0,t}$ and $p_{1,t}$ where each edge is associated with a node at next level, i.e., at level $i+1$, $p_{0,t}$ and $p_{1,t}$ are associated with $(i+1,p_{0,t}(j))$ and $(i+1,p_{1,t}(j))$ respectively for $t\in[L]$ and $i\in[w]$. BP starts at $(1,1)$ node and stops at (i,L) node, namely that $(1,1):\overset{\mathbf{x}}{\to}(i,L)$, $\mathbf{x}\in\{0,1\}^{L-1}$ is a boolean permutation. Hence, we associate the initial ciphertext $\mathbf{c}$ with the node $(1,1)$ of the BP, and the re-encryption key $\mathbf{X}_{var(t)}$ and $\mathbf{G}-\mathbf{X}_{var(t)}$ to the edges, then compute the BP on a permutation $rk=\{\mathbf{X}_{var(t)},\mathbf{G}-\mathbf{X}_{var(t)}\}^L$ as input, where we set rk is a sequence of re-encryption key, i.e., $(1,1):\overset{rk}{\to}(1,L)$. Actually, in our scheme, we first use the bit-decompose function Bit to decompose the secret key. Upon encrypting each bit of secret key, we collect all of these encrypted bits and assemble them into the re-encryption rk by using the Extend algorithm. We stress that rk is the sum of bit encryption of secret key by Extend algorithm.

4.1 Our Construction: Multi-Hop IBPRE via BP

In this subsection, we also adopt the same Setup, KeyGen, Enc and Eval algorithms and do not repeat here. We just focus on the ReKeyGen, ReEnc, and Dec algorithms. More details are as follows:

- $rk^{(i\to j)}\leftarrow\mathsf{ReKeyGen}(\mathsf{params},mpk,sk_i,id^{(i)},id^{(j)})$:
  In order to generate re-encryption key $rk^{(i\to j)}$ which is from player P_i to P_j, we encrypt the input side secret key (i.e., player P_i) in bit-by-bit manner according to the GSW encryption algorithm, perform the following steps:
  1) Firstly, samples a random matrix $\mathbf{R}^{(i\to j)}$ from $\{-1,0,1\}^{m\times(2n+1)\ell_q}$ and a random vector $\mathbf{z}_j\in\{-1,0,1\}^{1\times(2n+1)\ell_q}$, then computes

  $$\vec{\mathcal{S}}[i]=\left[\frac{\mathbf{u}^T\mathbf{R}^{(i\to j)}+\mathsf{Bit}_k(sk_i)\mathbf{G}+\mathbf{z}_j}{\mathbf{A}_{id^{(j)}}\cdot\mathbf{R}^{(i\to j)}}\right],$$

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

where Bit is bit decompose operation and $\mathsf{Bit}_k(sk_i)$ denotes the k-th bit of sk for $k \in [2n \cdot \ell_q]$;

2) Constructs a concatenation matrix $\vec{\mathcal{S}} = [\vec{\mathcal{S}}[1], \cdots, \vec{\mathcal{S}}[\ell_q]]$ by reassembling all of $\vec{\mathcal{S}}[i]$;

3) Invokes the Extend algorithm of Mukherjee and Wicks [43] to generate the matrix $\mathbf{M}^{(i\to j)} := \mathsf{Extend}\big(\vec{\mathcal{S}}, (\mathbf{u}, \mathbf{A}_{id^{(i)}}, \mathbf{A}_{id^{(j)}})\big) \in \mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$ which satisfies $\mathbf{M}^{(i\to j)} = \sum_i^{\ell_q} \vec{\mathcal{S}}[i]$. Actually, $\mathbf{M}^{(i\to j)}$ can be viewed as

$$\mathbf{M}^{(i\to j)} = \begin{bmatrix} \mathbf{u}^T\mathbf{R}^{(i\to j)} + [1, -\mathbf{e}_i^T]\mathbf{G} + \mathbf{z}_j \\ \mathbf{A}_{id^{(j)}}^T \cdot \mathbf{R}^{(i\to j)} \end{bmatrix};$$

4) Samples a random matrix $\mathbf{R}^{(j)} \leftarrow \{0,1\}^{m\times m}$ and computes $\mathbf{N}^{(j)} := (\mathbf{u} \mid \mathbf{A}_{id^{(j)}})^T \cdot \mathbf{R}^{(j)} \in \mathbb{Z}_q^{(2n+1)\times m}$. We stress that, $(\mathbf{u} \mid \mathbf{A}_{id^{(j)}}) \cdot \begin{pmatrix} \frac{1}{-\mathbf{e}_j} \end{pmatrix} = 0$.

5) Outputs the re-encryption key $rk^{(i\to j)} := \mathbf{K}^{(i\to j)} = (\mathbf{M}^{(i\to j)}, \mathbf{N}^{(j)})$.

- $\mathbf{c}_L \leftarrow \mathsf{ReEnc}\big((rk^{(0\to 1)}, \cdots, rk^{(L-1\to L)}), \mathbf{c}_0\big)$: In order to achieve *multi-hop re-encryption* by BP, i.e., we convert the augmented NAND circuits into BP. Hence, we proceed a BP homomorphically as follows,

  1) **Initiation:** For readability, without the loss of generality, we only consider $\mathbf{K}^{(i\to j)} := \mathbf{M}^{(i\to j)}$. We also omit the subscript, and abbreviate $rk^{(i\to j)}$ to rk_j ($\mathbf{K}^{(i\to j)}$ to $\mathbf{K}_j$), i.e., $rk_1 := \mathbf{K}_1, \cdots, rk_L := \mathbf{K}_L$. Then, we set $i = 0$, $\mathbf{c}_0 = 2^{\ell_q-1}\mathbf{t}_1$, and the state vector $\vec{\mathbf{w}}_0 := (2^{\ell_q-1}\mathbf{t}_1, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0})$, where we denote $\mathbf{t}_1 := [1, 0, \cdots, 0]^T \in \mathbb{Z}_q^{1\times nN}$. Next, we initiate the on-the-fly variant of Barrington's theorem $\mathsf{BPOTF}^{\mathsf{Pred}_\mathcal{C}}$ and invoke the predecessor function $\mathsf{Pred}_\mathcal{C}$ which takes the label i of a gate as input and outputs the next layer of BP, i.e., $\Big(\big(\gamma_{t,1,0}, \cdots, \gamma_{t,5,0}\big), \big(\gamma_{t,1,1}, \cdots, \gamma_{t,5,1}\big), var(t)\Big)$. We stress that $\mathsf{BPOTF}^{\mathsf{Pred}_\mathcal{C}}$ was denoted by Brakerski et al. [45], we adopt this definition and re-present it in Preliminaries. In more detail:

  2) **Iterative Step:** For every step $t = 1, \cdots, L$ it proceeds as follows:

  1). Computes the constants $\gamma_{t,i,0}$ and $\gamma_{t,i,1}$ by running $\mathsf{BPOTF}^{\mathsf{Pred}_\mathcal{C}}$ to obtain the next layer of the branching programs

  $$\Big(\big(\gamma_{t,1,0}, \cdots, \gamma_{t,5,0}\big), \big(\gamma_{t,1,1}, \cdots, \gamma_{t,5,1}\big), var(t)\Big)$$

  2). For every $i = 1, \cdots, 5$ it homomorphically computes the encryption of next state:

  $$\begin{aligned} \mathbf{w}_{t,i} \quad = \quad & (\mathbf{G} - \mathbf{K}_{vat(t)}) \cdot \mathbf{G}^{-1}(\mathbf{w}_{t-1,\gamma_{t,i,0}}) \\ & + \mathbf{K}_{vat(t)} \cdot \mathbf{G}^{-1}(\mathbf{w}_{t-1,\gamma_{t,i,1}}) \end{aligned}$$

  where $\mathbf{K}_{var(t)} := \sum \vec{\mathcal{S}}_{var(t)}$ is the sum encryption of the secret key $\vec{\mathcal{S}}_{var(t)}$. Finally, outputs the ciphertext $\mathbf{c}_L := \mathbf{w}_{L,1}$.

- $\mathsf{Dec}(\mathsf{params}, sk_L, \mathbf{c}_L)$: There also exist two cases.

  - Decryption at input side, which is the same as single-hop homomorphic IBPRE and outputs the results of $\langle \mathbf{c}, \mathbf{s} \rangle \pmod q$,

  $$\langle \mathbf{c}, \mathbf{s} \rangle := \mu \cdot 2^{\ell_q-1} + \mathbf{sx} \pmod q,$$

  If $\mathsf{noise}_{\mathbf{s},\mu}(\mathbf{c}) := \|\mathbf{sx}\| < q/8$, then sets $\mu = 1$ and otherwise sets $\mu = 0$. Outputs μ.

  - Decryption at output side, *L-level decryption algorithm*, after L-hop re-encryption from player $i \in [L]$ to player $j \geq i+1$. In this setting, re-encryption ciphertext is $\mathbf{c}_j := \mathbf{K}^{(i\to j)} \cdot \mathbf{G}^{-1}(\mathbf{c}_i)$, hence, computes and outputs the results of $\langle \mathbf{c}_j, \mathbf{s}_j \rangle \pmod q$. More concretely, after L-hop re-encryption from player $i := L-1$ to player $j := L$, the re-encryption ciphertext is $\mathbf{c}_L := \mathbf{K}^{((L-1)\to L)} \cdot \mathbf{G}^{-1}(\mathbf{c}_{(L-1)})$. Hence the decrypter computes and outputs the results of $\langle \mathbf{c}_L, \mathbf{s}_L \rangle \pmod q$, i.e., $\langle \mathbf{c}_L, \mathbf{s}_L \rangle = \mu 2^{\ell_q-1} + \mathbf{error}$, if $\|\mathbf{error}\| \leq 1/8$, then outputs $\mu = 1$ and otherwise $\mu = 0$.

4.2 Correctness and Security

In this subsection, we analyze the correctness and security of multi-hop homomorphic IBPRE. We note that, The correctness of decryption at input side can be obtained easily from the single-hop homomorphic IBPRE, i.e., Lemma 3.12. Thus we omit further details. Below, we analyze the correctness of decryption at output side.

Lemma 4.1 (Multi-Hop Correctness). *We can say the multi-hop homomorphic* IBPRE *scheme with short ciphertexts is correct after L-hop along with $t = 0, 1, \cdots, L$ and $i \in [5]$ if the following holds* $\mathsf{noise}_{sk_L,\mathbf{v}_t[i]}(\mathbf{w}_{t,i}) < 2t\big((n+1)\ell_q\big)^2 \cdot B$.

Proof: We first consider $\mathsf{noise}(\mathbf{w}_{0,i}) = 0$, since we setup $\vec{\mathbf{w}_0}$ is just a messages without noise. Assume that the hypothesis holds for $t' < t$, then it holds that for $t' = t-1$, namely that $\mathsf{noise}_{sk_{t-1},\mathbf{v}_t[i]}(\mathbf{w}_{t-1,\gamma_{t,i}}) = (t-1) \cdot \big((2n+1)\ell_q\big)^2 \cdot B$. Then, we prove it by induction on step t, by definition of $\mathbf{w}_t$, we obtain:

$$\begin{aligned} & \mathsf{noise}_{sk_L,\mathbf{v}_t[i]}(\mathbf{w}_{t,i}) \\ = \quad & \mathsf{noise}_{sk_L,\mathbf{v}_t[i]}\Big((\mathbf{G} - \mathbf{X}_{vat(t)}) \cdot \mathbf{G}^{-1}(\mathbf{w}_{t-1,\gamma_{t,i,0}}) \\ & + \mathbf{K}_{vat(t)} \cdot \mathbf{G}^{-1}(\mathbf{w}_{t-1,\gamma_{t,i,1}})\Big) \\ = \quad & (1 - x_{vat(t)}) \cdot \mathsf{noise}_{sk_L,\mathbf{v}_t[i]}\Big(\mathbf{w}_{t-1,\gamma_{t,i,0}}\Big) \\ & + 2(2n+1)\ell_q \cdot \mathsf{noise}_{sk_L,rk_{var(t)}}\Big(\mathbf{K}_{vat(t)}\Big) \\ & + x_{vat(t)} \cdot \mathsf{noise}_{sk_L,\mathbf{v}_t[i]}\Big(\mathbf{w}_{t-1,\gamma_{t,i,1}}\Big) \\ \leq \quad & \max\Big\{\mathsf{noise}_{sk_L,\mathbf{v}_t[i]}\Big(\mathbf{w}_{t-1,\gamma_{t,i,0}}\Big), \\ & \mathsf{noise}_{sk_L,\mathbf{v}_t[i]}\Big(\mathbf{w}_{t-1,\gamma_{t,i,1}}\Big)\Big\} \\ & + 2(2n+1)\ell_q \cdot \mathsf{noise}_{sk_L,rk_{var(t)}}\Big(\mathbf{K}_{vat(t)}\Big) \\ \leq \quad & 2(t-1)\big((2n+1)\ell_q\big)^2 \cdot B + 2\big((2n+1)\ell_q\big)^2 \cdot B \\ \leq \quad & 2t\big((2n+1)\ell_q\big)^2 \cdot B \end{aligned}$$

Moreover, after the L-hop iteration, and combining it with Lemma 4.1, we have that $\mathsf{noise}_{sk_L,\mathbf{v}_L[1]}(\mathbf{w}_{L,1}) < L \cdot 2t\big((2n+1)\ell_q\big)^2 \cdot B < \frac{q}{8}$, where $L = 4^d$ and d is circuit depth. This completes the proof. □

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

Theorem 4.2 (Security). *The multi-hop homomorphic* IBPRE *is IND-ID-CPA-secure if the single-hop homomorphic* IBPRE *under decision-*LWE *assumption is IND-ID-CPA-secure.*

Proof: The proof is the same as the proof of single-hop IBPRE except that we need repeat L times $\mathsf{Game}1, \cdots, \mathsf{Game}5$. Below, we sketch the proof.

1) Firstly, for each hop, the challenger $\mathcal{C}$ first picks $\mathbf{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[d]\times[\xi]})$ as $y_0, \{y_{i,j}\}_{(i,j)\in[d]\times[\xi]} \in \mathbb{Z}_q$ for $(i,j) \in [d] \times [\xi]$. For future convenience, we define a function $F_{\mathbf{y}}(id)$ at Corollary 3.7 and $F_{\mathbf{y}}(id)$ satisfying the following condition

$$F_{\mathbf{y}}(id^*) = 0 \bigwedge F_{\mathbf{y}}(id_1) \neq 0 \bigwedge \cdots \bigwedge F_{\mathbf{y}}(id_Q) \neq 0,$$

for the challenge identity id^* and $\mathcal{A}$ has made key extraction queries identity $id_1, \cdots, id_Q$, $Q < q$.
2) Secondly, as described in Game_2 and Game_3, for each hop, we consider the way of generation of mpk.
   - Considering the distributions of mpk $\left(\mathbf{A}, \mathbf{AR}_0 + y_0\mathbf{G}, \{\mathbf{AR}_{i,j} + y_{i,j}\mathbf{G}\}_{(i,j)\in[d]\times[\xi]}\right)$ and a uniform distribution $\left(\mathbf{A}, \mathbf{B}_0, \{\mathbf{B}_{i,j}\}_{(i,j)\in[d]\times[\xi]}\right)$ are statistically indistinguishable via leftover hash lemma, where $\mathbf{B}_0, \{\mathbf{B}_{i,j}\}_{(i,j)\in[d]\times[\xi]} \leftarrow \mathbb{Z}_q^{n\times m}$.
   - Consider the user-specific matrix $[\mathbf{A}, \mathcal{H}(id)]$, where $\mathcal{H}(id) = \mathbf{AR}_{id} + F_{\mathbf{y}}(id)\mathbf{G}$. If $F_{\mathbf{y}}(id) = 0$ and aborts, otherwise, we apply the leftover hash lemma to show that the user-specific matrix is indistinguishable from a uniform over $\mathbb{Z}_q^{m\times 2n}$.
3) Thirdly, as described in Game_4, for each hop, the challenger $\mathcal{C}$ first checks whether $F_{\mathbf{y}}(id) = 0$ and aborts, otherwise, $\mathcal{C}$ computes $\mathbf{R}_{id}$ via the function $\mathsf{TrapEval}$ as the definition of $\mathbf{R}_{id}$, then, $\mathcal{C}$ generates $\mathbf{e} \leftarrow \mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{id}, F_{\mathbf{y}}(id), \mathbf{u}, \mathbf{T}_{\mathbf{G}}, \sigma)$ as the secret key. We argue that the output distribution of $\mathsf{SampleRight}$ is indistinguishable from the sample over $D^{\mathbf{u}}_{(\mathbf{A}|\mathcal{H}(id)),\sigma}$ via $\mathbf{R}_{id}$ (Corollary 3.7) and the choice of σ by Lemma 2.11. We stress that, $\mathsf{SampleLeft} \approx_c D^{\mathbf{u}}_{(\mathbf{A}|\mathcal{H}(id)),\sigma}$.
4) Fourthly, for each hop, the challenger changes the way of the generation of $\mathbf{M}^{(i\to j)}$ by sampling from the distribution $\mathbb{Z}_q^{(2n+1)\times(2n+1)\ell_q}$ rather than by computing. In this setting, we use LWE assumption to show that the rk is indistinguishable from a uniform.
5) Fifthly, for each hop, we use the leftover hash lemma to show that the original ciphertext $\mathbf{c} = (\mathbf{u}, \mathbf{A}_{id})^T \cdot \mathbf{r} + \mu(2^{\ell_q-1}\mathbf{t}_1) + \mathbf{x} \pmod q$ at input side is indistinguishable from a uniform vector over $\mathbb{Z}_q^{(2n+1)\times 1}$;
6) Lastly, after 1-hop re-encryption, we have the ciphertexts $\mathbf{c}_2 := \mathbf{X}^{(1\to 2)} \cdot \mathbf{G}^{-1}(\mathbf{c}_1)$ at output side. Hence we use the leftover hash lemma to show that $\mathbf{c}_2$ is indistinguishable from a uniform vector over $\mathbb{Z}_q^{(2n+1)\times 1}$. Similarly, $\mathbf{c}_3 := \mathbf{X}^{(2\to 3)} \cdot \mathbf{G}^{-1}(\mathbf{c}_2)$ is indistinguishable from a uniform. Repeating L times in this way, i.e., after L-hop re-encryption, we get the ciphertext $\mathbf{c}_L := \mathbf{w}_{L,1}$ which is indistinguishable from a uniform vector over $\mathbb{Z}_q^{(2n+1)\times 1}$.

This completes the sketch of the proof. □

5 Conclusion

In this paper, we propose an efficient multi-hop homomorphic IBPRE scheme via BP. We give a comparison result of Chandran et al.'s PRE scheme [11], Singh et al.'s IBPRE scheme [14], Yamada's IBE scheme [7] and ours scheme in Table 2, where the parameters $m \geq n\log q + \lambda, n$ are the dimensions of the vectors, t is the length of encrypted messages, and $\ell_q = \lceil \log q \rceil$.

As is shown in Table 2, the length of the secret key, ciphertext, and re-encryption key of our scheme are better than Singh et al.'s IBPRE scheme [14]. Importantly, our scheme supports homomorphic operation and multi-hop re-encryption. Moreover, the parameters of our scheme are the same as the scheme of those of Yamada's scheme [7] except the re-encryption key. We stress that there are many follow-up works and optimizations of lattice-based PRE. Hence we just compare some closely related works with our scheme. We leave many works for future.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No.61472097).

References

[1] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, "Secure and efficient cloud data deduplication with randomized tag," *IEEE Trans. Inf. Foren. Secur.*, vol. 12, no. 3, pp. 532–543, 2017.
[2] H. Yan, J. Li, J. Han, and Y. Zhang, "A novel efficient remote data possession checking protocol in cloud storage," *IEEE Trans. Inf. Foren. Secur.*, vol. 12, no. 1, pp. 78–88, 2017.
[3] K. Liang, X. Huang, F. Guo, and J. K. Liu, "Privacy-preserving and regular language search over encrypted cloud data," *IEEE Trans. Inf. Foren. Secur.*, vol. 11, no. 10, pp. 2365–2376, 2016.
[4] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proc. STOC 2005*, pp. 84–93.
[5] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. STOC 2008*. ACM, pp. 197–206.
[6] C. Gentry *et al.*, "Fully homomorphic encryption using ideal lattices." in *Proc. STOC 2009*, vol. 9, no. 2009, pp. 169–178.
[7] S. Yamada, "Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters," in *Proc. EUROCRYPT 2016 Part II*, pp. 32–62.
[8] K. Xagawa, "Cryptography with lattices," Theses, Tokyo Institute of Technology, 2010. [Online]. Available: http://xagawa.net/pdf/2010Thesis.pdf
[9] Y. Aono, X. Boyen, L. Wang *et al.*, "Key-private proxy re-encryption under lwe," in *Proc. INDOCRYPT2013*, pp. 1–18.
[10] E. Kirshanova, "Proxy re-encryption from lattices," in *Proc. PKC 2014*, pp. 77–94.
[11] N. Chandran, M. Chase, F.-H. Liu, R. Nishimaki, and K. Xagawa, "Re-encryption, functional re-encryption, and multi-hop re-encryption: A framework for achieving obfuscation-based security and instantiations from lattices," in *Proc. PKC 2014*, pp. 95–112.
[12] R. Nishimaki and K. Xagawa, "Key-private proxy re-encryption from lattices, revisited," *IEICE Trans.*, vol. 98, no. 1, pp. 100–116, 2015.
[13] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proc. ACNS 2007*, pp. 288–306.
[14] K. Singh, C. P. Rangan, and A. K. Banerjee, "Lattice based identity based unidirectional proxy re-encryption scheme," in *Proc. SPACE*.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2740720, IEEE Access

Table 2
The Parameters Comparison of Some important PRE, IBE IBPRE and Ours Scheme

Scheme	Msg	$\|mpk\|$(or $\|pk\|$)	$\|sk\|$	$\|\mathrm{Ct}\|$($\|\mathrm{Re\text{-}Ct}\|$)	$\|rk\|$
Chandran et al. [11]	1	$m(n+1)\log q$	$n\log q$	$(n+1)\log q$	$n\ell_q(n+1)\log q$
Singh et al. [14]	t	$mn\log q$	$(m+m\ell_q)t\log q$	$(m+t)\log q$	$(m\ell_q+t)^2\log q$
Yamada [7]	1	$m(n+1)\log q$	$(2m+1)\log q$	$(2n+1)\log q$	✗
Our scheme	1	$m(n+1)\log q$	$(2m+1)\log q$	$(2n+1)\log q$	$(2n+1)\big((2n+1)\ell_q+m\big)\log q$

- $\|\mathrm{Ct}\|$: the length of the ciphertext;
- $\ell_q := \lceil \log q \rceil$;
- $\|\mathrm{Re\text{-}Ct}\|$: the length of the re-encrypted ciphertext;

[15] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," in *Proc. FOCS 2011*, pp. 97–106.
[16] Z. Li, C. Ma, G. Du, and O. Weiping, "Dual lwe-based fully homomorphic encryption with errorless key switching," in *Proc. ICPADS 2016*, pp. 1169–1174.
[17] X. Fan and F.-H. Liu, "Various proxy re-encryption schemes from lattices." *IACR Cryptology ePrint Archive, Report 2016/278*. [Online]. Available: http://eprint.iacr.org/2016/278
[18] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Proc. EUROCRYPT 2012*, pp. 700–718.
[19] Y. Ishai and A. Paskin, "Evaluating branching programs on encrypted data," in *Proc. TCC 2007*, pp. 575–594.
[20] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. EUROCRYPT1998*, pp. 127–144.
[21] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Proc. CRYPTO 1984*, pp. 10–18.
[22] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proc. CCS 2007*, pp. 185–194.
[23] J. Shao and Z. Cao, "Cca-secure proxy re-encryption without pairings," in *Proc. PKC 2009*, pp. 357–376.
[24] J. Weng, Y. Zhao, and G. Hanaoka, "On the security of a bidirectional proxy re-encryption scheme from pkc 2010," in *Proc. PKC 2010*, pp. 284–295.
[25] R. Lindner and C. Peikert, "Better key sizes (and attacks) for lwe-based encryption," in *Proc. CT-RSA 2011*, pp. 319–339.
[26] C. Ma, J. Li, and W. Ouyang, "A homomorphic proxy re-encryption from lattices," in *Proc. ProvSec 2016*, pp. 353–372.
[27] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Proc. CRYPTO 2013*, pp. 75–92.
[28] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proc. CRYPT 2001*, pp. 213–229.
[29] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (h) ibe in the standard model," in *Proc. EUROCRYPT2010*, pp. 553–572.
[30] J. Chen and H. Wee, "Fully,(almost) tightly secure ibe and dual system groups," in *Proc. CRYPTO 2013*, pp. 435–460.
[31] J. Zhang, Y. Chen, and Z. Zhang, "Programmable hash functions from lattices: short signatures and ibes with small key sizes," in *Proc. CRYPTO 2016, Part III*, pp. 303–332.
[32] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy, "Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits," in *Proc. EUROCRYPT 2014*, pp. 533–556.
[33] Y. Ren, D. Gu, S. Wang, and X. Zhang, "Hierarchical identity-based proxy re-encryption without random oracles," *International Journal of Foundations of Computer Science*, vol. 21, no. 06, pp. 1049–1063, 2010.
[34] J. Shao and Z. Cao, "Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption," *Information Sciences*, vol. 206, pp. 83–95, 2012.
[35] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *Proc. ASIACCS 2009*, pp. 276–286.
[36] T. Matsuo, "Proxy re-encryption systems for identity-based encryption," in *Proc. PAIRING 2007*, pp. 247–267.
[37] J. Li, X. Tan, X. Chen, D. S. Wong, and F. Xhafa, "Opor: Enabling proof of retrievability in cloud computing with resource-constrained devices," *IEEE Trans. Cloud Computing*, vol. 3, no. 2, pp. 195–205, 2015.
[38] H. Wang, D. He, and S. Tang, "Identity-based proxy-oriented data uploading and remote data integrity checking in public cloud," *IEEE Trans. Inf. Foren. Secur.*, vol. 11, no. 6, pp. 1165–1176, 2016.
[39] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical gapsvp," in *Proc. CRYPTO2012*, pp. 868–886.
[40] V. Lyubashevsky, "Lattice signatures without trapdoors," in *Proc. EUROCRYPT 2012*, pp. 738–755.
[41] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem," in *Proc. STOC 2009*, pp. 333–342.
[42] C. Peikert and B. Waters, "Lossy trapdoor functions and their applications," in *Proc. STOC 2008*, pp. 187–196.
[43] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key fhe," in *Proc. EUROCRYPT 2016, Part II*, pp. 735–763.
[44] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," in *Proc.ITCS2012*, pp. 309–325.
[45] Z. Brakerski and R. Perlman, "Lattice-based fully dynamic multi-key fhe with short ciphertexts," in *Proc. CRYPTO2016*, pp. 190–213.
[46] Z. Brakerski and V. Vaikuntanathan, "Lattice-based fhe as secure as pke," in *Proc. ITCS 2014*, pp. 1–12.
[47] D. A. Barrington, "Bounded-width polynomial-size branching programs recognize exactly those languages in nc1," *J. Comput. Syst. Sci.*
[48] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *Proc. CRYPTO 2011*. Springer, pp. 505–524.
[49] Z. Li, C. Ma, E. Morais, and G. Du, "Multi-bit leveled homomorphic encryption via dual.lwe based," in *Proc. Inscrypt 2016*, pp. 221–242.

Zengpeng Li Currently, he is pursuing his Ph.D. degree at Harbin Engineering University, China. His primary research interests are in cryptography, protocol, and information security. In particular, his research focus is lattice-based cryptography. He is a student member of IEEE, CCF, IEICE and CACR.

Chunguang Ma received his Ph.D. degree in cryptography from College of Computer Science and Technology, Beijing University of Posts and Telecommunications, China, in 2005. He is currently a professor at College of Computer Science and Technology, Harbin Engineering University. His main research interests include cryptography and information security, in particular, cryptographic protocols.

Ding Wang Currently, he is a Postdoc. at Peking University, China. He has been involved in the community as a TPC member and a reviewer for over 50 international conferences and journals. His works appear in prestigious venues like ACM CCS, IEEE/IFIP DSN, ESORICS and IEEE TDSC, and three of them are selected as "ESI highly cited papers". His research interests include cryptography and system security.

2169-3536 (c) 2017 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.