Fast Proxy Re-Encryption for Publish/Subscribe Systems *

Yuriy Polyakov[†1,2], Kurt Rohloff[‡1], Gyana Sahu[§1] and Vinod Vaikuntanthan[¶2]

[1]New Jersey Institute of Technology, Newark NJ, USA
[2]Massachusetts Institute of Technology, Cambridge MA, USA

May 14, 2017

Abstract

We develop two IND-CPA-secure multi-hop unidirectional Proxy Re-Encryption (PRE) schemes by applying the Ring-LWE (RLWE) relinearization approach from the homomorphic encryption literature. Unidirectional PRE is ideal for secure publish-subscribe operations where a publisher encrypts information using a public key without knowing upfront who the subscriber will be and what private key will be used for decryption. The proposed PRE schemes provide a multi-hop capability, meaning that when PRE-encrypted information is published onto a PRE-enabled server, the server can either delegate access to specific clients or enable other servers the right to delegate access. Our first scheme (which we call NTRU-ABD-PRE) is based on a variant of the NTRU-RLWE homomorphic encryption scheme. Our second and main PRE scheme (which we call BV-PRE) is built on top of the Brakerski-Vaikuntanathan (BV) homomorphic encryption scheme and relies solely on the RLWE assumption. We present an open-source C++ implementation of both schemes and discuss several algorithmic and software optimizations. We examine parameter selection tradeoffs in the context of security, runtime/latency, throughput, ciphertext expansion, memory usage, and multi-hop capabilities. Our experimental analysis demonstrates that BV-PRE outperforms NTRU-ABD-PRE both in single-hop and multi-hop settings. The BV-PRE scheme has a lower time and space complexity than existing IND-CPA-secure lattice-based PRE schemes, and requires small concrete parameters, making the scheme computationally efficient for use on low-resource embedded systems while still providing 100 bits of security. We present practical recommendations for applying the PRE schemes to several use cases of ad-hoc information sharing for publish-subscribe operations.

*Partially sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Army Research Laboratory (ARL) under Contract Numbers W911NF-15-C-0226 and W911NF-15-C-0233. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the Department of Defense or the U.S. Government. Project partially sponsored by the National Security Agency under Grant H98230-15-1-0274. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either express or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

[†]polyakov@njit.edu
[‡]rohloff@njit.edu Corresponding Author
[§]grs22@njit.edu
[¶]vinodv@mit.edu

1 Introduction

Modern encryption algorithms are powerful tools to achieve confidentiality in information systems. In modern encryption systems, data is encrypted so it can be decrypted only by users with a specific decryption key. For private-key (also called symmetric) encryption systems such as AES, the encryption and decryption keys are the same, and every effort must be made to prevent leaking the keys to adversaries. For public-key (also called asymmetric) encryption systems such as RSA, one uses a paired public key and secret key so that data, once encrypted with a public key, can only be decrypted with its corresponding secret key. If an adversary obtains the public key, she will not be able to decrypt the data.

An unfortunate limitation of most modern private-key and public-key cryptosystems is that once data is encrypted, it is generally impossible to transform the encrypted data so that it can be decrypted with a different key without sharing a common secret or decrypting the data as an intermediate step. This ability to "switch" the key that data is encrypted under can be considered as a way to delegate access to a new, previously unintended recipient. Without this access delegation capability it is difficult to securely use encryption technologies in publish-subscribe scenarios where publishers encrypt data that is posted to an information broker server, and subscribers can later (asynchronously) access and decrypt the data (especially when the publisher does not know the subscriber).

Conventional techniques for publish-subscribe scenarios require trust either between publisher and subscriber or between publisher and broker. For instance, the publisher secret key can be encrypted using a subscriber public key and can then be transmitted to the subscriber. This approach based on direct coordination between the publisher and subscriber is an inconvenience which limits application to where this is feasible and complicates fine-grained delegation of data which has already been published to the broker.

Proxy Re-Encryption (PRE) provides an approach to circumvent these limitations. PRE enables subscribers to receive and decrypt encrypted data that they are intended to receive without ever directly coordinating decryption keys with the publisher of the data. To do this, PRE securely transforms the ciphertexts from the public key under which sensitive data is encrypted to ciphertexts decryptable under the subscriber secret key without full decryption or unallowed access to sensitive data.

Concretely, the mode of PRE operation considered in this paper is as follows. Publishers use their public keys to encrypt data. A PRE re-encryption process uses a special re-encryption key that is generated without interaction between publishers and subscribers to convert publisher encrypted data so it can be decrypted only by an intended subscriber. The re-encryption key is generated from the publisher's secret key and the subscribers public key. The re-encryption key is generated by the publisher, or by a trusted proxy of the publisher such as an access control policy authority. The *re-encryption* process happens at a PRE server. If properly designed, implemented and configured, the confidentiality of information brokered by the PRE server is preserved even if an adversary observes all of the encrypted messages sent to and from the PRE server, the public keys used by the publisher, and the re-encryption keys used by the PRE server. Data is never in the clear during the re-encryption operation. As such, information confidentiality is maintained even if the PRE server is captured or compromised. Information confidentiality is also maintained for information not stored in the clear by the publishers if the publishers are captured or compromised. This holds even if an adversary were to manipulate the PRE server manufacturing process with compromised circuitry. More formally, encrypted ciphertexts are IND-CPA-secure, which means

they do not reveal even a single bit of the information, even given all the publisher public keys and re-encryption keys.

Proxy Re-Encryption, as we have described it, is also called a *unidirectional PRE* scheme, and was defined in [19, 3]. Henceforth, when we refer to proxy re-encryption, we mean a *unidirectional* scheme. In this paper we describe two new IND-CPA-secure Proxy Re-Encryption (PRE) schemes and their implementations where the PRE functionality is provided using the RLWE relinearization procedure. The first scheme (NTRU-ABD-PRE) is based on the NTRU encryption scheme with RLWE modifications [35] where the NTRU immunity constraint against subfield lattice attacks is applied to set the distribution parameter for NTRU key generation [1]. The second scheme (BV-PRE) is based on the BV homomorphic encryption scheme [9] and relies only the RLWE assumption.

As opposed to other known approaches to PRE, lattice encryption approaches, such as ours, are generally considered post-quantum [30, 34], that is, potentially secure against attacks even from adversaries with practical quantum computing devices in addition to adversaries with classical computing devices [27]. Our goal is to provide a software PRE capability that can support high-throughput pub-sub information sharing without direct interactions between publishers and subscribers. We provide experimental evaluation of software implementations of our PRE scheme to evaluate its security, scalability and performance.

BV-PRE outperforms NTRU-ABD-PRE in both single-hop and multi-hop settings, and is provably secure under the RLWE assumption, in contrast to the NTRU variant which is proven secure under a less well-studied variant of the so-called NTRU assumption. BV-PRE scales well with the number of hops due to a relatively small additive noise growth provided by the BV scheme and RLWE relinearization procedure. BV-PRE has small ciphertext modulus and ring dimension requirements: successful decryption after re-encryption can be achieved using a 17-bit ciphertext modulus and ring dimension of 512 (for at least 100 bits of security). This translates to submillisecond encryption/decryption runtime and re-encryption runtime of under 5 ms.

When compared to the LWE-based PRE lattice schemes recently proposed in [2, 20, 16, 33], our BV-PRE scheme provides key sizes and ciphertext expansion factors as good as or better than the key sizes of any other lattice-based PRE schemes, and lower time complexity than any other LWE-based scheme.

Paper Organization. In Section 2 we review related results on encrypted computing and PRE. In Section 3 we discuss the the high-level concept and performance and security tradeoffs of PRE. In Sections 4 and 5 we describe the proposed lattice-based PRE cryptosystems. In Section 6 we discuss parameter selection for both schemes to provide practical secure computing on commodity computing hardware. In Section 7 we describe the overall software architecture of the library to support the end-to-end encrypted application. In Section 8 we discuss experimentation and evaluation of our design and implementation. In Section 9 we present practical use cases for the proposed PRE cryptosystem. We conclude the paper with a discussion of our insights and future work in Section 10.

2 Related Work

The notion of Proxy Re-Encryption (PRE) was introduced in the work of Blaze, Bleumer and Strauss [5], where they also presented a construction based on the El-Gamal encryption scheme. Their construction was a *bidirectional* proxy re-encryption scheme in that a re-encryption key can

be used to translate encrypted data from the publisher encryption to subscriber encryption but also in reverse, from the subscriber encryption to publisher encryption. In contrast, in this work, we focus on *unidirectional* proxy re-encryption that provides tighter control on which ciphertexts can be re-encrypted and to whom.

Unidirectional PRE schemes were first proposed in [19, 3]. The scheme in [3] is based on the decisional bilinear Diffie-Hellman (DBDH) assumption. The schemes in [19, 3] are single-hop proxy re-encryption schemes, meaning that a re-encrypted ciphertext cannot be re-encrypted further, to a third party.

Also related is the work in [11] which provides a multi-hop bidirectional scheme based on bilinear maps. *Multi-hop* re-encryption schemes are important in applications where re-encryption needs to be applied multiple times, for example where information needs to be brokered in multiple steps from publisher to subscriber. We refer the reader to Sections 9.1 and 9.2 for a discussion of applications.

Our approach to PRE is based on two common ring variants of lattice-based homomorphic encryption schemes, with the PRE functionality provided using the RLWE relinearization procedure of Brakerski and Vaikuntanathan [9]. The first scheme (NTRU-ABD-PRE) is built on top of the NTRU encryption scheme [18] with RLWE modifications [35] where the NTRU immunity constraint against subfield lattice attacks is applied to set the distribution parameter for NTRU key generation (c.f. [1]). The second scheme (BV-PRE) is based on the BV homomorphic encryption scheme [9] and relies only on the RLWE assumption.

Both variants of ring homomorphic encryption schemes have been used to build the fully homomorphic encryption (FHE) capability [9, 7, 17, 23, 6, 15]. FHE schemes are encryption schemes that allow anyone to run computations over encrypted data without decrypting the data.

It is at times difficult to establish direct comparisons between encryption schemes, even with similar computational hardness underpinnings. Following [12, 21, 30, 36], we use the standard "root Hermite factor" δ as the primary measure of the concrete security of RLWE-based encryption schemes, for a given set of parameters, where a smaller δ provides more security. Experimental evidence [12] suggests that $\delta = 1.007$ would require roughly 2^{40} core-years on recent Intel Xeon processors to break. We set the configuration parameters to attain δ of just less than 1.006 for each of the schemes for our parameter and key size comparisons, and for our later experimental analyses. The root Hermite factor parameter setting we use of $\delta < 1.006$ arguably provides at least 100 bits of security [12, 21, 30, 36].

Whereas our BV-PRE scheme provides sub-millisecond runtimes for optimal parameter settings for encryption and decryption operations and millisecond-order runtimes for re-encryption, the experimental results of [3] are in the ranges of 3 to 9 ms (for 256 bits of security) and 8 to 27 (for 512 bits of security) for these same operations. However, the experimental results of the non-lattice-based work in [3] are shown for 256 and 512 bits of security rather than approximately 100 bits in our case. Our estimates using equation (7) in [17] show that $\delta \approx 1.0034$ and $\delta \approx 1.002$ correspond to 256 and 512 bits of securiy, respectively. These values of δ increase the minimum ring dimension for 256 bits of security to 1024 and for 512 bits of security to 2048, while keeping the bit width approximately the same. This implies that the runtime is roughly doubled when one goes from 100 bits of security to 256 and then doubles again when going from 256 to 512 bits of security, which suggests that our runtimes are comparable to those reported in [3].

An independent work of [31] proposes and implements a IND-CPA-secure proxy re-encryption scheme based on the NTRU encryption scheme with RLWE modifications [35], which they label

Table 1: Parameter configuration and key size comparison of LWE-based IND-CPA-secure PRE schemes for normalized conditions, i.e., plaintext modulus $p = 2$, relinearization window $r = 1$, message length $l = n$ (in the LWE scheme) with comparable security (root Hermite factor δ is under 1.006; this bound corresponds to approximately 100 bits of security.)

Scheme	Parameters for Secure Configuration			Key Sizes for Secure Operation, KB			Asymptotic Key Sizes		
	δ	n	k	**sk**	**pk**	**rk**	**sk**	**pk**	**rk**
BV-PRE	1.0051	512	17	1.06	2.13	36.1	nk	$2nk$	$2nk^2$
NTRU-ABD-PRE	1.0054	1024	35	4.38	4.38	153	nk	nk	nk^2
PS-NTRUReEncrypt [31]	1.0037	2048	47	11.8	11.8	11.8	nk	nk	nk
IND-CPA-LWE [33]	1.0042	450	14	346	692	9,690	n^2k	$2n^2k$	$2n^2k^2$

as PS-NTRUReEncrypt. This PRE scheme relies on a variant of NTRU assumption. The PS-NTRUReEncrypt construction is a bidirectional PRE scheme, whereas ours is unidirectional (see Section 9 for why unidirectional schemes are critical to our applications). The runtimes reported in [31] are of the order of one second. The authors [31] also propose and implement another bidirectional (more efficient but not IND-CPA-secure) scheme called NTRUReEncrypt with runtimes of the order of one millisecond. However, NTRUReEncrypt is not directly comparable to our schemes in security as its security relies on an ad-hoc new assumption. It is therefore unclear how to set key-sizes for this scheme, and hence, we will not discuss this scheme further in this paper. We note, however, that our RLWE-based BV-PRE scheme achieves a comparable, even better, performance with the added provable security guarantee based on the relatively well-studied RLWE problem.

Several LWE-based PRE lattice schemes have recently been proposed in [2, 20, 16, 33]. The schemes presented in [20, 16] are based on a Regev-style encryption, which can be regarded as an extension of the CCA-secure public key encryption scheme developed in [28]. The schemes developed in [2, 33, 16] are based on a public key encryption scheme formulated in [21]. [16] shows an implementation of a IND-CPA-secure multi-hop scheme. The most efficient implemented variant [33], which we label as IND-CPA-LWE, is similar to BV-PRE but relies on the LWE assumption (instead of ideal lattices and RLWE assumption). This scheme is also unidirectional and supports multiple hops of re-encryptions.

Table 1 shows the comparison of parameter selections, resulting concrete secure key sizes and asymptotic key sizes for the following LWE-based IND-CPA-secure PRE schemes: NTRU-ABD-PRE, BV-PRE, PS-NTRUReEncrypt [31], and the IND-CPA-secure LWE scheme [33]. We base these comparisons on roughly equivalent functionality and security configurations. For notational simplicity we define $k = \lfloor \log_2 q + 1 \rfloor$, the number of bits required to represent the ciphertext modulus q. For the concrete parameters in the table, we set the ring dimension n (referred to as the lattice security parameter n in the case of the LWE scheme) and ciphertext modulus q for each of the schemes for a single-hop use case for plaintext modulus $p = 2$ to ensure that the security parameter $\delta < 1.006$. Note that for the PS-NTRUReEncrypt and IND-CPA-LWE schemes we use a tighter bound on the root Hermite factor δ due to the parameter selection decisions made in the papers we cited for the schemes.

In comparison with prior lattice-based PRE schemes:

- The key sizes and ciphertext expansion factors of BV-PRE and NTRU-ABD-PRE are as good as or better than the key sizes of the other lattice-based PRE schemes.
- The ciphertext expansion factor of NTRU-ABD-PRE and PS-NTRUReEncrypt is k and $2k$ for BV-PRE and CP-LWE. However, NTRU-ABD-PRE and PS-NTRUReEncrypt require higher parameter values, which automatically increases space requirements for the secret and public keys and encryption/decryption execution time.
- Noise grows multiplicatively with the number of re-encryption hops in the case of NTRU-ABD-PRE (at most two hops are supported). BV-PRE, IND-CPA-LWE, and PS-NTRUReEncrypt can support up to 100 hops without significantly increasing the ring dimension (lattice security parameter) and ciphertext bit length due to additive noise growth.
- Although the re-encryption space and time complexity for PS-NTRUReEncrypt is lower, this scheme is bidirectional and does not support the same security use cases as BV-PRE and IND-CPA-LWE.
- IND-CPA-LWE has much higher space requirements (an additional factor of n in the size of all keys) as compared to BV-PRE, which limits its applicability to embedded systems.

The above analysis implies that BV-PRE is more efficient for Pub/Sub systems than all existing lattice-based PRE schemes.

We also provide a high-level theoretical evaluation of the performance of our schemes in comparison with other identified recent lattice-based IND-CPA-secure PRE schemes. Rather than base this initial comparison on experimental runtime performance, we compare performance in terms of the computational operations which are generally the lower-level computational building blocks provided by math libraries and hardware accelerators which implementations are built from. In particular, we couch our evaluation of theoretical performance in terms of the number of slightly higher-level polynomial operations, inclusive of Number Theoretic Transforms (NTT's), Vector Products (VP's), Matrix Vector Products (MVP's) and Bit-decomposed Matrix Vector Products (BMVP). This allows us to present complexity comparisons independent of the specific differences in implementation libraries that might be used to experimentally compare these schemes. A table with comparisons for encryption, re-encryption and decryption operations of our schemes and PS-NTRUReEncrypt [31] and IND-CPA-LWE [33] schemes can be seen in Table 2. In this table, the short-hand notation of $(k{+}1)$ NTT + $2k$ VP for the cell corresponding to the re-encryption complexity for BV-PRE is used to indicate that the re-encryption operations requires $(k{+}1)$ calls to an NTT operation and $2k$ calls to a VP operation. As a matrix-vector product of $n \times n$ by n generally has a higher complexity than Number Theoretic Transform (NTT), which is $O(n \log n)$, the runtime of BV-PRE is expected to be smaller for comparable values of ring dimension (lattice security parameter) and ciphertext modulus bit-width than IND-CPA-LWE.

In summary, the relation of our work to the prior work is as follows.

- Constructions of PRE based on bilinear maps are either single-hop unidirectional [3] or multi-hop bidirectional [11], whereas our scheme is multi-hop unidirectional. As noted in [11], constructing a multi-hop unidirectional PRE scheme using bilinear maps is an open problem.

Table 2: Theoretical complexity comparison of LWE-based IND-CPA-secure PRE schemes for normalized conditions, i.e., plaintext modulus $p = 2$, relinearization window $r = 1$, message length $l = n$ (in the LWE scheme).

Scheme	Runtime/Latency		
	Enc	**ReEnc**	**Dec**
BV-PRE	1 NTT + 2 VP	(k+1) NTT + 2k VP	1 NTT + 1 VP
NTRU-ABD-PRE	1 NTT + 1 VP	(k+1) NTT + k VP	1 NTT + 1 VP
PS-NTRUReEncrypt [31]	1 NTT + 1 VP	1 VP	1 NTT + 1 VP
IND-CPA-LWE [33]	2 MVP	2 BMVP	1 MVP

The practical execution times of our BV-PRE scheme (order of one millisecond), which supports dozens of hops without significant performance degradation, are comparable to those of bilinear map-based constructions.

- The BV-PRE scheme has a lower time and space complexity than existing IND-CPA-secure lattice-based PRE schemes.

3 Proxy Re-Encryption

3.1 Workflow

The basic usage of Proxy Re-Encryption is shown in Figure 1. We assume a slightly more general model for PRE operations where a Policy Authority operates as a proxy for Alice to generate Alice's public key and generate re-encryption keys to control who can decrypt information encrypted by Alice. It is also possible for Alice to be her own Policy Authority. The high-level operational flow of this key management infrastructure is as follows:

1. The policy authority generates public and secret key pairs for publishers such as Alice. These keys are designated as pk_A and sk_A, respectively. This key generation operation nominally occurs prior to deployment, or just as publishers need to send information to a PRE server.

2. Prior to deployment, the policy authority sends the publisher Alice public key pk_A. The publisher (and possibly multiple publishers) uses this public key to encrypt ciphertexts $c_A = \mathsf{Enc}(m, pk_A)$ they send to the PRE server. The policy authority retains the secret key sk_A in case it needs to access information encrypted by the publisher.

3. When a subscriber needs to receive information from the PRE server, the subscriber Bob sends his public key (pk_B) to the policy authority.

4. The policy authority uses the publisher secret key (sk_A) and the subscriber public key (pk_B) to generate a re-encryption key (rk_{AB}). This re-key generation could occur prior to deployment or just as a subscriber needs to receive information.

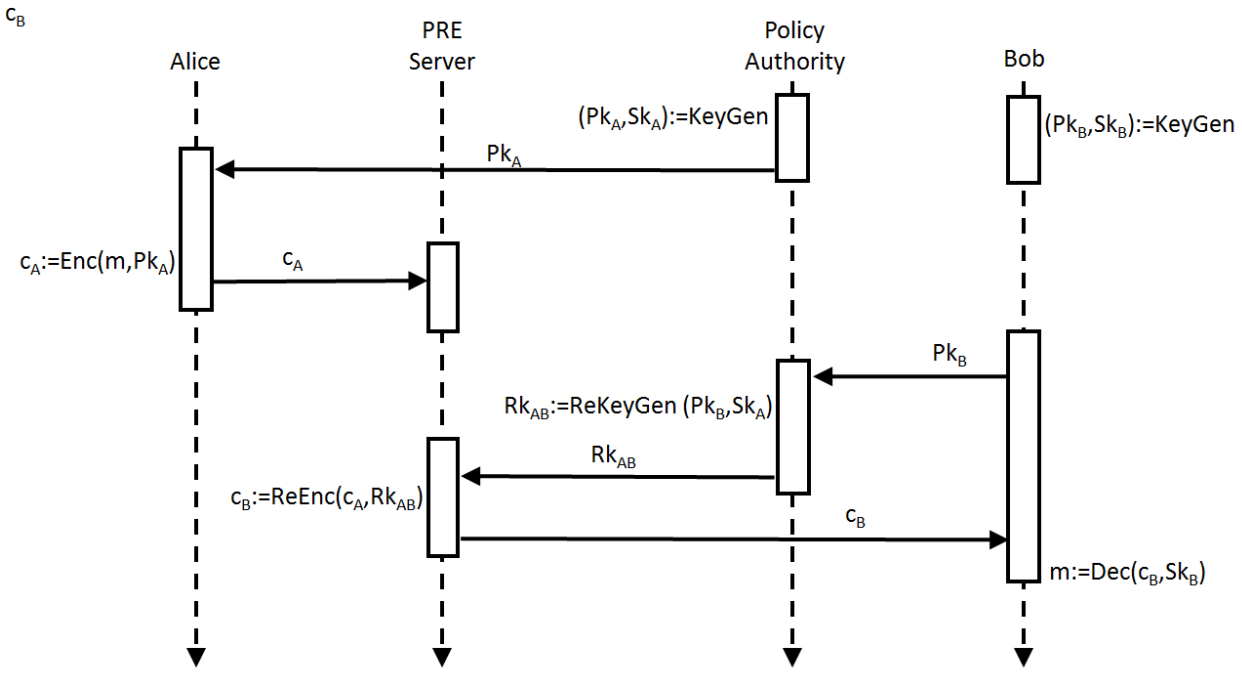

Figure 1: Proxy Re-Encryption Functional Key Management and Interaction Workflow

5. The policy authority sends the re-encryption key to the PRE server.
6. The PRE server re-encrypts ciphertext so it can be decrypted by Bob.
7. Bob receives ciphertext and decrypts it using its secret key sk_B.

An important aspect of this key management infrastructure is that PRE pushes trust from the publisher to the policy authority and computational effort and bandwidth requirements to the PRE server. The policy authority determines who can share information and the PRE server uses information access policies to determine what subset of information from the publisher should be sent to the subscriber. The publisher and subscriber, who generally have the lowest computational capability in mobile applications, require the lowest computational effort and only need to maintain single keys, thus simplifying mobile deployments.

3.2 Syntax of Non-Interactive PRE

The workflow depicted in Figure 1 can only be supported by non-interactive PRE schemes, which require that re-encryption keys are generated using Bob's public key and Alice's private key. In this case, direct interaction between Bob and Alice can be avoided.

A non-interactive scheme includes algorithms (ParamsGen, KeyGen, ReKeyGen, Enc, ReEnc, Dec), described as follows:

- ParamsGen(λ): returns public parameters pp corresponding to security parameter λ;
- KeyGen(pp, λ): returns public-secret key pairs (pk, sk);

- $\mathsf{ReKeyGen}(pp, sk_i, pk_j)$: returns the re-encryption key $rk_{i\to j}$;
- $\mathsf{Enc}(pp, pk, m)$: encrypts message m using pk and returns the ciphertext;
- $\mathsf{ReEnc}(pp, rk_{i\to j}, c_i)$: transforms a ciphertext c_i of party i into a ciphertext c_j that party j can decrypt;
- $\mathsf{Dec}(pp, sk, c)$: recovers message m.

3.3 IND-CPA Security of PRE Schemes

Our security definition is a variant of the one postulated by Ateniese, Fu, Green and Hohenberger [3]. While Ateniese et al. [3] considered the notion of single-hop PRE schemes, both us and [33] consider multi-hop PRE schemes. We remark that *the distinction between single-hop and multi-hop PRE is one of correctness, not security.* We state the definition below.

Definition 1 (IND-CPA Security)**.** *We consider the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ which proceeds in two phases.*

***Phase 1**:*

- *$\mathcal{C}$ generates public parameters $pp \leftarrow \mathsf{ParamsGen}(\lambda)$ and gives them to $\mathcal{A}$.*
- *Uncorrupted key generation: $\mathcal{C}$ generates $(pk, sk) \leftarrow \mathsf{KeyGen}(pp, \lambda)$ and gives pk to $\mathcal{A}$ upon request. $\mathcal{A}$ can request polynomially many such pk. Let Γ_H be the set of honest public keys (also called honest entities).*
- *Corrupted key generation: $\mathcal{C}$ generates $(pk, sk) \leftarrow \mathsf{KeyGen}(pp, \lambda)$ and gives (pk, sk) to $\mathcal{A}$ upon request. $\mathcal{A}$ can request polynomially many such (pk, sk). Let Γ_C be the set of corrupted public keys (also called corrupted entities).*

The adversary can issue a polynomial number of these queries, in arbitrary order.

***Phase 2**:*

- *Re-encryption key generation: The adversary submits a directed acyclic* re-encryption graph *$G = (V, E)$ where the vertex set V is the set of all uncorrupted keys the adversary requested in Phase 1. $\mathcal{A}$ is given all the re-encryption keys $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(pp, sk_i, pk_j)$ where $(i, j) \in E$.*

  We remark that the adversary can generate by herself all re-encryption keys $rk_{i\to j}$ where $i \in \Gamma_C$, since she knows the secret keys sk_i. On the other hand, she is not allowed to obtain $rk_{i\to j}$ where $i \in \Gamma_H$ and $j \in \Gamma_C$ as that could allow her to decrypt the challenge ciphertext simply by performing a sequence of re-encryptions.

  We also note that this mechanism already allows the adversary to obtain re-encryptions of any ciphertext that she wishes. To obtain the re-encryption of an adversarially chosen string c_i from the public key pk_i to pk_j, she simply uses the re-encryption key $rk_{i\to j}$ that she obtained and runs the honest re-encryption procedure. Thus, there is no need to handle a separate re-encryption query.
- *Challenge: $\mathcal{A}$ submits (i^*, m_0, m_1). $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and then returns $c_{i^*} \leftarrow \mathsf{Enc}(pp, pk_{i^*}, m_b)$. This is done only once, and it is required that $i^* \in \Gamma_H$.*

$\mathcal{A}$ finally outputs $b' \in \{0,1\}$ as a guess of b. Define $\mathcal{A}$'s advantage as

$$\boldsymbol{Adv}_{\mathcal{A}}^{cpa}(\lambda) = \left| Pr\left[b' = b\right] - \frac{1}{2} \right|.$$

The PRE scheme is IND-CPA-secure if this advantage is negligible for all poly-time adversaries $\mathcal{A}$.

A few remarks about this definition are in order.

First, assume that the proxy obtains a (unidirectional) re-encryption key from user Alice to user Bob. The security definition above implies that even if the proxy (who has the re-encryption key) and Alice collude, they cannot break the security of Bob's encryption.

Secondly, note that if the proxy and Bob collude, they can decrypt Alice's ciphertexts, *by definition*. This is simply because the proxy can use the re-encryption key to turn Alice's ciphertext into Bob's ciphertext (for the same message) and then proceed to use Bob's secret key to decrypt. In essence, this means that the proxy and Bob together have a decryption circuit for Alice. (We do not attempt to define the notion of allowing this collusion to obtain only a "weak secret key" as in [3]).

Third, we note that stronger definitions are possible in that they can allow for chosen ciphertext decryption queries as considered in the work of [11]. One way to capture such attacks in the framework of our definition is to allow the adversary to request for re-encryption keys from uncorrupted public keys to corrupted ones, except that he cannot ask for a path of re-encryption keys from the challenge public key to a corrupted public key. We do not pursue this line of definitions in this work.

Fourth, we note that our IND-CPA definition does not explicitly handle re-encryption queries by the adversary, namely where the adversary queries with a tuple (i, j, c) and obtains the result of the re-encryption algorithm applied to $rk_{i\to j}$ and c. The reason we do not do this explicitly is that the adversary can simulate by herself the execution of such a query by first asking our re-encryption key generation oracle for $rk_{i\to j}$ and using it to re-encrypt c by herself. Since the pairs of keys for which the adversary is permitted to make re-encryption queries is the same as the ones between which she can obtain a re-encryption key, this omission is without loss of generality.

Finally, we note that the single-hop scheme of [3] is secure in a stronger IND-CPA sense than the above, since they can handle re-encryption graphs that contain directed cycles. On the other hand, the security proof of [33] appears to only handle our acyclic IND-CPA definition.

4 PRE Cryptosystem with NTRU Key Generation and RLWE Relinearization (NTRU-ABD-PRE)

The first PRE scheme proposed in this paper is based on the NTRU encryption scheme [18] with RLWE modifications [35]. The NTRU immunity constraint against subfield lattice attacks is used to set the distribution parameter for NTRU key generation [1]. The subfield lattice attacks allow the adversary to reduce the ring dimension of the affected cryptosystems for certain parameter regimes and solve the Shortest Vector Problem for n = 512 or lower [1, 13].

4.1 NTRU-RLWE Encryption Scheme

The scheme is parameterized using the following quantities:

- security parameter (root Hermite factor) δ [12],
- ciphertext modulus q,
- ring dimension n,
- B_k-bounded discrete Gaussian key distribution χ_k over the polynomial ring $R = \mathbb{Z}[n]/\langle x^n + 1\rangle$ with distribution parameter σ_k (subscript k refers to *key* distribution),
- B_e-bounded discrete Gaussian error distribution χ_e with distribution parameter σ_e (subscript e refers to *error* distribution),
- empirically selected assurance measure α to minimize the number of bits needed to represent q (introduced for better performance).

In this work we focus on the case of power-of-2 cyclotomic polynomials where n is a power of 2 for multiple reasons. For one, the case of power-of-2 cyclotomics leads to much simpler and more efficient implementations of number theoretic transforms used to support ring operations. Further, the computational hardness underlying security assumptions for these cases is better understood as compared to the case of arbitrary cyclotomics [35].

We support a plaintext space of $\mathcal{M} = \{0, 1, \ldots, p-1\}^n$, where $p \geq 2$ is the plaintext modulus. All operations on ciphertexts are performed in the ring $R_q \equiv R/qR$. Each coefficient in the polynomials is represented in the range $\left\{-\left\lfloor \frac{q}{2} \right\rfloor, ..., \left\lfloor \frac{q}{2} \right\rfloor\right\}$.

The scheme includes the following operations:

- $\mathsf{ParamsGen}(\lambda)$: Choose positive integers q and n. Return $pp = (q, n)$.
- $\mathsf{KeyGen}(pp, \lambda)$: Sample polynomials $f', g \leftarrow \chi_k$ and set $f := pf' + 1$ to satisfy $f \equiv 1 \,(\mathrm{mod} p)$. If f has no modular multiplicative inverse in R_q, then re-sample f'. Set the public key pk and private key sk:
$$sk := f \in R,\ pk := h = pg\, f^{-1} \in R_q.$$
- $\mathsf{Enc}\,(pp, pk = h, m \in \mathcal{M})$: Sample polynomials $s, e \leftarrow \chi_e$. Compute the ciphertext:
$$c := hs + pe + m \in R_q.$$
- $\mathsf{Dec}\,(pp, sk = f,\ c)$: Compute the ciphertext error $b := f \cdot c \in R_q$. Output $m' := b \,(\mathrm{mod} p)$.

The scheme is correct as long as there is no wrap-around modulo q. Indeed,

$$b = f \cdot c = f\,(h\,s + pe + m) = pgs + pfe + fm$$

and if the value of b does not wrap around modulo q, then

$$m' = pgs + pfe + fm = fm = m \ (\mathrm{mod} p).$$

To derive the correctness constraint for decryption, we note that the coefficients in g, s cannot exceed B_k as they are generated by a B_k-bounded discrete Gaussian distribution χ_k. Analogously, the coefficients in f cannot exceed $pB_k + 1$ and coefficients in e cannot exceed B_e, yielding

$$\|b\|_\infty = \|pgs + pfe + fm\|_\infty < 2np^2 B_k B_e.$$

Here, we assume that $B_k, B_e \gg 1$, which is true for all practical scenarios of this scheme. To guarantee correct decryption of the ciphertext, coefficients in b should not exceed $q/2$ leading to the following correctness constraint:

$$q > 4np^2 B_k B_e. \tag{1}$$

When $\sigma > \omega(\log n)$, the bound B_i can be expressed as $\sigma_i\sqrt{n}$, where $i \in \{k, e\}$ and 2^{-n+1} is the probability that a coefficient generated using discrete Gaussian distribution exceeds the bound B_i [29, 23]. To obtain less conservative estimated bounds for noise, we introduce an assurance measure $\alpha < n$ corresponding to the probability of $2^{-\alpha+1}$ that a coefficient of a discrete Gaussian polynomial exceeds the bound B_i (the choice of a specific practical value of α is validated using an empirical analysis of decryption correctness for a large sample of plaintexts). In this case, the bounds B_k and B_e can be expressed as $\sigma_k\sqrt{\alpha}$ and $\sigma_e\sqrt{\alpha}$, respectively.

The constraint (1) was derived for the worst-case scenario where both B_i-bounded polynomials may simultaneously take the upper (or lower) bound values for all coefficients in the polynomials of dimension n. As the coefficients of polynomials generated by the discrete Gaussian distribution are taken from a relatively large sample of size n (where n is at least 512), we can apply the Central Limit Theorem (CLT) to derive a lower (average-case) bound for q.

If we examine a product of two B_k-bounded polynomials g and s in the ring R_q, we observe that each coefficient in g is multiplied by the mean of coefficients in s (followed by modulo reduction). This implies that each coefficient in $g \cdot s$ is bounded by $n\sigma_k\sigma_{kn}\alpha$, where σ_{kn} is the standard deviation of the mean expressed as $\sigma_{kn} = \sigma_k n^{-1/2}$. After simplification, the bound for $g \cdot s$ can be expressed as $\sqrt{n}\sigma_k^2\alpha$ instead of the original worst-case bound $n\sigma_k^2\alpha$. Therefore, this technique allows one to replace each occurrence of n (corresponding to a polynomial multiplication) with $\sqrt{n}$. Applying this logic to the worst-case constraint for the encryption scheme, we obtain the following average-case correctness constraint:

$$q > 4\sqrt{n}p^2 B_k B_e. \tag{2}$$

It should be noted that the effective probability associated with assurance measure α, i.e., $2^{-\alpha+1}$, gets significantly reduced for a product of two discrete Gaussians. This further justifies the use of an assurance measure much smaller than n.

4.2 Security of NTRU-RLWE Encryption Scheme

This general NTRU-RLWE encryption scheme can be instantiated for three different ranges of distribution parameter σ_k, giving us security from different computational assumptions [35, 23, 1]. The scheme can be proven secure based on the NTRU and RLWE assumptions for these different parameter regimes. Here, the NTRU problem is to distinguish between the following two distributions: a polynomial f/g with f and g sampled from distribution χ_k (assuming g is invertible over R_q) and a polynomial h sampled uniformly at random over R_q.

The first variant [35] is based on the RLWE assumption. The public key (polynomial f/g) distribution was shown to be *statistically indistinguishable* from uniform random distribution for $\Phi_m(x) = x^n + 1$ when $\sigma_k = \omega\left(q^{1/2}\right)$ [35]. This allowed the authors to rely solely on the RLWE assumption to prove semantic security of the encryption scheme. This logic was applied to show that the Stehlé-Steinfeld scheme defined by operations $\mathsf{KeyGen}, \mathsf{Enc}$, and Dec and constraint $\sigma_k = \omega\left(q^{1/2}\right)$ is IND-CPA secure [35].

Though based solely on the RLWE assumption, the original Stehlé-Steinfeld scheme is impractical for proxy re-encryption or any homomorphic encryption scheme requiring at least one

multiplication of two polynomials generated from the distribution χ_k. In this case, the correctness inequality for q would never hold as we would have $B_k^2 \propto \sigma_k^2 = \omega(q)$ on the right hand side of the expression, i.e., $q > \omega(q)$.

For practical reasons, the constraint $\sigma_k = \omega\left(q^{1/2}\right)$ was suggested to be replaced with a smaller value corresponding to the error distribution χ_e by arguing that the resulting Decisional Small Polynomial Ratio (DSPR) problem is secure against all known practical attacks [23]. This assumption was recently invalidated for some parameter ranges by two subfield lattice attacks [1, 13], which are able to reduce the ring dimension of the affected cryptosystems and solve the Shortest Vector Problem for n = 512 or lower.

Albrecht et al. [1] proposed a new practical constraint for σ_k based on the immunity of NTRU to subfield lattice attacks, conjecturing that the Stehlé-Steinfeld proof may be extended to this case:

$$\sigma_k > \left(\frac{2q}{n\pi e}\right)^{1/4}. \tag{3}$$

Our proxy re-encryption scheme, referred to as NTRU-ABD-PRE, uses this constraint. In contrast to the original Stehlé-Steinfeld scheme, this scheme supports ReKeyGen, ReEnc, and homomorphic indexing and multiplication operations.

To meet the RLWE security requirements of the encryption scheme, we use the inequality derived in [17], namely,

$$n \geq \frac{\log_2(q/\sigma_e)}{4\log_2(\delta)}. \tag{4}$$

4.3 Single-Hop Re-Encryption Scheme

The PRE scheme introduces a new configurable parameter, relinearization window r, and two new operations (in addition to ParamsGen, KeyGen, Enc, and Dec):

- ReKeyGen $(pp, sk = f, pk = h^*)$: For every $i = 0, 1, .., \lfloor \log_2(q)/r \rfloor$, sample polynomials s_i and e_i, and compute γ_i

$$\gamma_i = h^* s_i + p e_i + f \cdot (2^r)^i \in R_q,\ rk := \gamma = \left(\gamma_0, \gamma_1, ..., \gamma_{\lfloor \log_2(q)/r \rfloor}\right).$$

- ReEnc $(pp, rk = \gamma, c)$: Compute the ciphertext

$$c' = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_i \cdot \gamma_i),$$

where $c_i := \{h \cdot s + pe + m\}_i$, $c = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} c_i \cdot (2^r)^i$.

Here, $rk = \gamma$ is the re-encryption key. The relinearization window r is used to decompose the ciphertext coefficients into base-2^r components c_i, thus substantially reducing the noise growth. Each c_i is represented as a polynomial in R_q with coefficients in the range between 0 and $2^r - 1$.

The PRE scheme is devised using a generalized version of the RLWE relinearization (bit decomposition) technique originally introduced for reducing the ciphertext error in homomorphic encryption [8, 23]. Consider a new set of keys: private key f^* and public key $h^* = pg^*(f^*)^{-1}$. The goal is to re-encrypt the ciphertext c using the public key h^* without decrypting the data.

To this end, we introduce a set of elements γ_i as

$$\gamma_i = h^* s_i + pe_i + f \cdot (2^r)^i \in R_q,$$

where $i = 0, 1, .., \lfloor \log_2 (q) / r \rfloor$. This set of elements, referred to as the re-encryption key, represents an encryption of all powers-of-2^r multiples of the secret key f under the public key h^*. The relinearization window was set to unity in [8, 23]. In this study, we consider a range of relinearization window values (powers of 2) to achieve a faster implementation of re-encryption. The vector $\gamma = \left(\gamma_0, \gamma_1, ..., \gamma_{\lfloor \log_2(q)/r \rfloor}\right)$ is public.

The ciphertext c is computed using the public key h:

$$c := hs + pe + m \in R_q.$$

For each window i of length r (in bits), we introduce $c_i := \{h \cdot s + pe + m\}_i$, and the ciphertext c can then be represented as

$$c = \sum_i c_i \cdot (2^r)^i.$$

The polynomial c' computed as

$$c' = \sum_i c_i \cdot \gamma_i$$

can be shown to represent an encryption of m under the new public key h^*.

Indeed,

$$f^* \cdot c' = f^* \cdot (\sum_i c_i \cdot \gamma_i) = \sum_i c_i \cdot (f^* \cdot \gamma_i) = p \sum_i c_i \cdot E_i + \sum_i c_i f^* f \cdot (2^r)^i = p \sum_i c_i \cdot E_i + f^* f\, c,$$

where $E_i = g^* s_i + f^* e_i$.

It can be seen that

$$f^* \cdot c' = f^* f\, c = m \,(\mathrm{mod} p)\,,$$

i.e., the decryption is correct, if the ciphertext error $f^* \cdot c'$ is not too large to wrap around q.

Considering that $\|c_i\|_\infty \leq 2^r - 1$, $\|E_i\|_\infty \leq nB_e\,(B_k + pB_k + 1)$, and

$$\|f^* f\, c\|_\infty \leq n^2\,(pB_k + 1)\,\{pB_e\,(B_k + pB_k + 1) + (pB_k + 1)\,(p - 1)\}\,,$$

we have

$$\begin{aligned} \left\|f^* c'\right\|_\infty &\leq pn\,(\lfloor \log_2 (q) / r \rfloor + 1)\,(2^r - 1)\,\{nB_e\,(B_k + pB_k + 1)\} \\ &\quad + n^2\,(pB_k + 1)\,(pB_e\,(B_k + pB_k + 1) + (pB_k + 1)\,(p - 1)\} \\ &< 2p^2 n^2 B_e B_k\,\{(2^r - 1)\,(\lfloor \log_2 (q) / r \rfloor + 1) + pB_k\}\,. \end{aligned}$$

To guarantee correct decryption of the ciphertext, $f^* \cdot c'$ should not exceed $q/2$ leading to the following worst-case correctness constraint:

$$q > 4p^2 n^2 B_k B_e\,\{pB_k + (2^r - 1)\,(\lfloor \log_2 (q) / r \rfloor + 1)\}\,.$$

Similar to the case of the encryption scheme, we can apply CLT to obtain the following average-case correctness constraint for the PRE scheme:

$$q > 4p^2 n B_k B_e\,\{pB_k + (2^r - 1)\,(\lfloor \log_2 (q) / r \rfloor + 1)\}\,. \tag{5}$$

4.4 Extension to Multiple Re-Encryption Hops

The presented re-encryption scheme can be generalized to support multiple re-encryption hops. Consider a new set of keys: private key f^{**} and public key $h^{**} = pg^{**}\left(f^{**}\right)^{-1}$. The goal is to re-encrypt the re-encrypted ciphertext c' devised in Section 4.3 using the public key h^{**} without decrypting the data.

Analogously to the case of single re-encryption, we introduce a set of elements

$$\gamma'_i = h^{**}s'_i + pe'_i + f^* \cdot \left(2^r\right)^i \in R_q,$$

where $i = 0, 1, .., \lfloor \log_2 (q) / r \rfloor$. The vector $\gamma' = \left(\gamma'_0, \gamma'_1, ..., \gamma'_{\lfloor \log_2(q)/r \rfloor}\right)$ is the re-encryption key to transform from $\{f^*, h^*\}$ to $\{f^{**}, h^{**}\}$.

The polynomial c'' computed as

$$c'' = \sum_i c'_i \cdot \gamma'_i$$

can be shown to represent an encryption of m under the new public key h^{**} as long as there is no wrap-around modulo q.

Indeed,

$$f^{**} \cdot c'' = \sum_i {c'}_i \cdot \left(f^{**} \cdot \gamma'_i\right) = p \sum_i {c'}_i \cdot E'_i + \sum_i {c'}_i f^{**} f^* \cdot \left(2^r\right)^i = p \sum_i c'_i \cdot E'_i + f^{**} f^* \, c',$$

where $E'_i = g^{**}s'_i + f^{**}e'_i$.

It can be easily shown that

$$f^{**} \cdot c'' = f^{**} f^* \cdot c' = f^{**} f^* f \cdot c = m \left(\mathrm{mod} p\right),$$

i.e., the decryption is correct, if the ciphertext error $f^{**} \cdot c''$ is not too large to wrap around q.

Applying the same procedure as for the first re-encryption, the correctness inequality after two re-encryptions can be expressed as

$$q > 4p^2 n B_k B_e \left\{\left(2^r - 1\right) \left(\lfloor \log_2 (q) / r \rfloor + 1\right)\right\}$$
$$+ 4p^2 n B_k B_e n^{0.5} \left(pB_k + 1\right) \left\{pB_k + \left(2^r - 1\right) \left(\lfloor \log_2 (q) / r \rfloor + 1\right)\right\}.$$

Considering that the first summand is at least by a factor of $n^{0.5}\left(pB_k + 1\right)$ (this value is larger than 2^{10} for all practical parameter ranges) smaller than the second summand, the correctness constraint can be rewritten as

$$q > n^{0.5} \left(pB_k + 1\right) \cdot 4p^2 n B_k B_e \left\{pB_k + \left(2^r - 1\right) \left(\lfloor \log_2 (q) / r \rfloor + 1\right)\right\}, \tag{6}$$

which implies that the second re-encryption increases the lower bound for q by a factor of $n^{0.5}\left(pB_k + 1\right)$.

After two re-encryption hops, the expression on the right hand side of (6) is $\Omega(B_k^3)$ and $B_k^3 \propto \sigma_k^3 \propto q^{3/4+\epsilon}$, where $\epsilon > 0$. This implies that NTRU-ABD-PRE does not support more than two re-encryption hops because in the case of three hops the right-hand side expression of (6) will reach $q^{1+\epsilon}$.

The effective value of assurance measure α corresponding to a given probability can be decreased for each extra step of re-encryption as long as the empirical evaluation of decryption correctness is performed.

4.5 IND-CPA Security

We will show that the NTRU-ABD-PRE scheme is IND-CPA secure in the sense of Definition 1.

As noted in section 4.2, the NTRU-ABD-PRE scheme is based on the NTRU and RLWE assumptions. Specifically, we use a variant of the NTRU assumption formulated in [1] to achieve the immunity of NTRU to subfield lattice attacks. We refer to this variant as NTRU-ABD with the formal definition as follows:

Definition 2. *The NTRU-ABD$_{n,q,\chi_k}$ problem is to distinguish between the following two distributions over ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$: a polynomial g/f with g and f sampled from distribution χ_k with $\sigma_k > \Theta(q^{1/4})$ (assuming g is invertible over R_q) and a polynomial h sampled uniformly at random over R_q.*

For the RLWE assumption, we use the Hermite normal form [23], which is defined as follows:

Definition 3. *For all $\lambda \in \mathbb{N}$, let $\phi(x) = \phi_\lambda(x) \in \mathbb{Z}[x]$ be a cyclotomic polynomial of degree $n = n(\lambda)$, let $q = q(\lambda) \in \mathbb{Z}$ be a prime number, let the ring $R \doteq \mathbb{Z}[x]/\langle \phi(x)\rangle$ and $R_q \doteq R/qR$, and let χ_e denote an error distribution over the ring R.*

The decision ring-LWE assumption RLWE$_{\phi,q,\chi_e}$ states that for any ℓ = poly(λ),

$$\{(a_i, a_i \cdot s + e_i)\}_{i\in[\ell]} \overset{c}{\approx} \{(a_i, u_i)\}_{i\in[\ell]},$$

where s and "error polynomials" e_i are sampled from the noise distribution χ_e, and a_i and u_i are uniformly random in R_q.

Albrecht *et al.* [1] conjectured that the Stehlé-Steinfeld IND-CPA proof [35], which was provided for $\sigma_k = \omega(q^{1/2})$, may be extended to this case assuming only RLWE. However, as it stands, the security of this scheme is based on RLWE as well as the NTRU-ABD assumption described above. We will assume that the NTRU-ABD assumption is stronger than RLWE and set the key-sizes accordingly.

We showed that the NTRU-ABD PRE scheme maintains *correctness* for only two hops, in the sense that once a ciphertext goes through more than two hops, it cannot be decrypted to the correct message any more. It is important to note that the "two-hopness" is a limitation on the correctness of the scheme, not its security. In particular, we will show below that the scheme is IND-CPA-secure in the sense of Definition 1 which is a notion of security for general, multi-hop PRE schemes.

Our proof for NTRU-ABD-PRE is similar to that of the IND-CPA-secure LWE scheme proposed in [33].

Theorem 1 (IND-CPA security of NTRU-ABD-PRE)**.** *Under the NTRU-ABD$_{n,q,\chi_k}$ and RLWE$_{\phi,q,\chi_e}$ assumptions, NTRU-ABD-PRE is IND-CPA-secure. Specifically, for a poly-time adversary $\mathcal{A}$, there exists a poly-time distinguisher $\mathcal{D}$ such that*

$$\boldsymbol{Adv}^{cpa}_{\mathcal{A}}(\lambda) \leq (\rho \cdot (Q_{rk} + Q_{re}) + N + 1) \cdot \mathsf{max}(\boldsymbol{Adv}^{NTRU\text{-}ABD_{n,q,\chi_k}}_{\mathcal{D}}(\lambda), \boldsymbol{Adv}^{RLWE_{\phi,q,\chi_e}}_{\mathcal{D}}(\lambda))$$

where Q_{rk} and Q_{re} are the numbers of re-encryption key queries and re-encryption queries, respectively; N is the number of honest entities; λ is the security parameter; $\rho := 1 + \lfloor \log_2 q/r \rfloor$.

Proof. We show that the NTRU-ABD-PRE scheme is IND-CPA-secure through a sequence of games.

Let Game_0 be an initial game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ with their interactions governed by Definition 1. For notational convenience, let us consider the case when $\Gamma_H = \{1, \ldots, N\}$ and $\Gamma_C = \{N+1, \ldots, M\}$ for some polynomial M. Furthermore, without loss of generality, let $1, 2, \ldots, N$ be the topological order dictated by the re-encryption graph, starting from the sinks to the sources, namely there are no edges from i to k if $i < k$. In more detail:

- The i-th key pair is defined as $sk_i := f_i \in R$, and $pk_i := h_i = pg_i f_i^{-1} \in R_q$, where $f_i = pf_i' + 1$ and $f_i', g_i \leftarrow \chi_k$.
- The re-encryption key from party i to party k is written as

$$rk_{i \to k} := (h_k \cdot s_{iku} + pe_{iku} + f_k \cdot (2^r)^u)_{u \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}},$$

  where s_{iku}, e_{iku} are generated by party i.
- The challenge ciphertext of message m_b related to party i^* is:

$$c^* := h^* \cdot s^* + pe^* + m_b \in R_q,$$

  where $b \in \{0, 1\}$ is the challenge bit, $s^*, e^* \leftarrow \chi_e$, and h^* is the challenge public key.

Let Game_k $(1 \le k \le N)$ be defined by considering the honest party $k \in \Gamma_H$. Game_k is identical to Game_{k-1} except for the following changes:

- When generating the k-th key pair, $h_k = p \cdot r_k^*$, where r_k^* is a randomly generated ring element rather than an NTRU-ABD sample.
- When answering the re-encryption key query (i, k): First, note that $i > k$ because of the topological ordering. The re-encryption key is expressed as

$$rk_{i \to k} := (p \cdot \gamma_{iku})_{u \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}},$$

  where γ_{iku} is freshly random.

Each Game_k is computationally indistinguishable from Game_{k-1} because of the NTRU-ABD and RLWE assumptions. First, $k \in \Gamma_H$ and therefore, there is no re-encryption "edge" from user k to any user in Γ_C. Additionally, and crucially, all the re-encryption keys (k, i) have already been replaced with uniformly random ring elements in the prior games. Consequently, the secret key f_k is used only in the form of fresh NTRU-ABD samples in its public key and in the form of fresh RLWE samples in the re-encryption keys (the latter assumes that the public key is indistinguishable from a random sample based on the the NTRU-ABD assumption). Thus, all these can be replaced by uniformly random ring elements by invoking the NTRU-ABD and RLWE assumptions. The security loss is proportional to the number of re-encryption key and re-encryption queries that user k was part of (an additional multiplicative factor $1 + \lfloor \log_2 q/r \rfloor$ is incurred in the security loss as each re-encryption key contains that many RLWE samples).

$\text{Game}_{\text{final}}$ is the same as Game_N except for the challenge ciphertext that is expressed as

$$c^* := p \cdot r^* + m_b \in R_q,$$

where r^* is a freshly random ring element in R_q. This is computationally indistinguishable from Game_N by the RLWE assumption (assuming that the public key is indistinguishable from a random sample based on the the NTRU-ABD assumption).

The last change guarantees that the challenge bit b is information-theoretically hidden from $\mathcal{A}$, and therefore, the advantage of the adversary in Game_1 is 0.

Putting all this together, we see that

$$\mathbf{Adv}^{cpa}_{\mathcal{A}}(\lambda) \leq (\rho \cdot (Q_{rk} + Q_{re}) + N + 1) \cdot \mathsf{max}(\mathbf{Adv}^{NTRU\text{-}ABD_{n,q,\chi_k}}_{\mathcal{D}}(\lambda), \mathbf{Adv}^{RLWE_{\phi,q,\chi_e}}_{\mathcal{D}}(\lambda))$$

where $\rho := 1 + \lfloor \log_2 q/r \rfloor$. This finishes our proof. □

5 PRE Cryptosystem with RLWE Key Generation and Relinearization (BV-PRE)

The second PRE scheme proposed in this paper, BV-PRE, is based on the Brakerski-Vaikuntanathan (BV) homomorphic encryption scheme [9]. The BV-PRE scheme relies only on the RLWE security assumption.

5.1 The Encryption Scheme

The basic encryption scheme comes from the work of Lyubashevsky, Peikert and Regev [24, 25] and Micciancio [26]. The scheme is parameterized using the following quantities:

- security parameter (root Hermite factor) δ [12],
- ciphertext modulus q,
- ring dimension n,
- B_e-bounded discrete Gaussian error distribution χ_e with distribution parameter σ_e,
- empirically selected assurance measure α to minimize the number of bits needed to represent q (introduced for better performance).

As in the case of NTRU-ABD-PRE, we work with the polynomial ring $R_q = \mathbb{Z}_q[n]/\langle x^n + 1\rangle$ and use a plaintext space of $\mathcal{M} = \{0, 1, \ldots, p-1\}^n$, where $p \geq 2$ is the plaintext modulus. Each coefficient in the polynomials is represented in the range $\left\{-\lfloor \frac{q}{2} \rfloor, ..., \lfloor \frac{q}{2} \rfloor\right\}$. We also introduce $\mathcal{U}_q$ as a discrete uniform random distribution over R_q.

The scheme includes the following operations:

- $\mathsf{ParamsGen}(\lambda)$: Choose positive integers q and n. Return $pp = (q, n)$.
- $\mathsf{KeyGen}(pp, \lambda)$: Sample polynomials $a \leftarrow \mathcal{U}_q$ and $s, e \leftarrow \chi_e$. Compute $b := a \cdot s + pe \in R_q$. Set the public key pk and private key sk:

$$sk := s \in R,\ pk := (a, b) \in R_q^2.$$

- $\mathsf{Enc}\left(pp, pk = (a, b), m \in \mathcal{M}\right)$: Sample polynomials $v, e_0, e_1 \leftarrow \chi_e$. Compute the ciphertext $c = (c_0, c_1) \in R_q^2$:

$$c_0 := b \cdot v + pe_0 + m \in R_q,\ c_1 := a \cdot v + pe_1 \in R_q.$$

- $\mathsf{Dec}\,(pp, sk = s,\ c)$: Compute the ciphertext error $t := c_0 - s \cdot c_1 \in R_q$. Output $m' := t\,(\mathrm{mod} p)$.

The scheme is correct as long as there is no wrap-around modulo q. Indeed,

$$\begin{aligned} t &= b \cdot v + pe_0 + m - s \cdot (a \cdot v + pe_1) = (a \cdot s + pe) \cdot v + pe_0 + m - s \cdot (a \cdot v + pe_1) \\ &= m + p\,(e \cdot v + e_0 - s \cdot e_1)\,. \end{aligned}$$

where all computations are done mod q. If the value of t does not wrap around modulo q, then

$$m' = m + p\,(e \cdot v + e_0 - s \cdot e_1) = m\ (\mathrm{mod} p).$$

To derive the correctness constraint for decryption, we note that the coefficients in s, v, e, e_0, e_1 cannot exceed B_e as they are generated by a B_e-bounded discrete Gaussian distribution χ_e. Applying the same procedure as for the NTRU-RLWE scheme followed by CLT, we obtain

$$\|t\|_\infty < 3\sqrt{n} p B_e^2.$$

Here, we assume that $B_e > 1$. To guarantee correct decryption of the ciphertext, coefficients in t should not exceed $q/2$, leading to the following correctness constraint:

$$q > 6\sqrt{n} p B_e^2. \tag{7}$$

5.2 Proxy Re-Encryption Scheme

The PRE scheme introduces three new operations (in addition to $\mathsf{ParamsGen}$, KeyGen, Enc, and Dec) in contrast to two needed for the PRE functionality in NTRU-ABD-PRE. In BV-PRE, the evaluation key generation is performed in two separate stages: $\mathsf{Preprocess}$ and $\mathsf{ReKeyGen}$. First, the owner of key s^* generates a set of "public" keys $(\beta_i, \beta_i \cdot s^* + pe_i)$ and then sends these keys to the policy authority, as displayed in Figure 1. After that, the proxy authority computes γ_i to generate the complete re-encryption key. This allows one to apply the same non-interactive PRE workflow as discussed in Section 3.

- $\mathsf{Preprocess}\,(pp, \lambda, sk^* = s^*)$: For every $i = 0, 1, .., \lfloor \log_2(q)/r \rfloor$, where r is the relinearization window, sample polynomials $\beta_i \leftarrow \mathcal{U}_q$ and $e_i \leftarrow \chi_e$ and compute

$$\theta_i^* = \beta_i \cdot s^* + pe_i \in R_q,$$

$$pk := (\beta_i, \theta_i^*)_{i \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}}\,.$$

- $\mathsf{ReKeyGen}\left(pp, sk = s, pk = (\beta_i, \theta_i^*)_{i \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}}\right)$:
For every $i = 0, 1, .., \lfloor \log_2(q)/r \rfloor$, compute γ_i and store them in re-encryption key rk

$$\gamma_i = \theta_i^* - s \cdot (2^r)^i \in R_q, rk := (\beta_i, \gamma_i)_{i \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}}\,.$$

- $\mathsf{ReEnc}\left(pp, rk = (\beta_i, \gamma_i)_{i \in \{0,1,\ldots \lfloor \log_2(q)/r \rfloor\}}, c\right)$: Compute the ciphertext $c' = (c_0', c_1')$ using the 2^r-base decomposition of ciphertext element c_1 of original ciphertext $c = (c_0, c_1)$

$$c_0' = c_0 + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_1^{(i)} \cdot \gamma_i),\ c_1' = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_1^{(i)} \cdot \beta_i),$$

where $c_1^{(i)} := \{a \cdot v + pe\}_i$ is the i_{th} "digit" of the base-2^r representation of c_1 and

$$c_1 = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} c_1^{(i)} \cdot (2^r)^i.$$

The ciphertext $c' = (c'_0, c'_1)$ can be shown to represent an encryption of m under the new key s^*. Indeed,

$$\begin{aligned} c'_0 - s^* c'_1 &= c_0 + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_1^{(i)} \cdot \gamma_i) - s^* \cdot \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_1^{(i)} \cdot \beta_i) \\ &= c_0 + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} \left(c_1^{(i)} \cdot \left\{ \beta_i \cdot s^* + pe_i - s \cdot (2^r)^i \right\} \right) - s^* \cdot \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (c_1^{(i)} \cdot \beta_i) \\ &= c_0 - s \cdot c_1 + pE_i, \end{aligned}$$

where $E_i = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} \left(c_1^{(i)} \cdot e_i \right)$.

It can be easily seen that

$$c_0 - s \cdot c_1 + pE_i \,(\mathrm{mod} p) = c_0 - s \cdot c_1 \,(\mathrm{mod} p) = m \,(\mathrm{mod} p).$$

The above analysis implies that the ciphertext noise term grows only by a small additive factor $p \left\| E_i \right\|_\infty$ after each re-encryption. $\left\| E_i \right\|_\infty$ can be expressed as

$$\left\| E_i \right\|_\infty < \sqrt{n} B_e \left(2^r - 1\right) \left(\lfloor \log_2 (q) / r \rfloor + 1 \right).$$

Therefore, the correctness constraint for d re-encryption hops can be written as

$$q > 2\sqrt{n} p B_e \left\{ 3B_e + d \cdot \left(2^r - 1\right) \left(\lfloor \log_2 (q) / r \rfloor + 1 \right) \right\}. \tag{8}$$

5.3 IND-CPA Security

We will show that the BV-PRE scheme is IND-CPA secure in the sense of Definition 1.

Theorem 2 (IND-CPA security of BV-PRE)**.** *Under the $RLWE_{\phi,q,\chi_e}$ assumption, BV-PRE is IND-CPA-secure. Specifically, for a poly-time adversary $\mathcal{A}$, there exists a poly-time distinguisher $\mathcal{D}$ such that*

$$\boldsymbol{Adv}_{\mathcal{A}}^{cpa}(\lambda) \leq (\rho \cdot (Q_{rk} + Q_{re}) + N + 1) \cdot \boldsymbol{Adv}_{\mathcal{D}}^{RLWE_{\phi,q,\chi_e}}(\lambda)$$

where Q_{rk} and Q_{re} are the numbers of re-encryption key queries and re-encryption queries, respectively; N is the number of honest entities; λ is the security parameter; ϕ is the cyclotomic polynomial defining the ring $R_q = \mathbb{Z}_q[x] / \langle \phi \rangle$ and $\rho := 1 + \lfloor \log_2 q / r \rfloor$.

Proof. We show that the BV-PRE scheme is IND-CPA-secure through a sequence of games.

Let Game_0 be an initial game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ with their interactions governed by Definition 1. For notational convenience, let us consider the case when $\Gamma_H = \{1, \ldots, N\}$ and $\Gamma_C = \{N+1, \ldots, M\}$ for some polynomial M. Furthermore, without loss of generality, let $1, 2, \ldots, N$ be the topological order dictated by the re-encryption graph, starting from the sinks to the sources, namely there are no edges from i to k if $i < k$. In more detail:

- The i-th key pair is defined as $sk := s_i \in R$, and $pk := (a_i, a_i \cdot s_i + pe_i) \in R_q^2$, where $s_i, e_i \leftarrow \chi_e$.
- The re-encryption key from party i to party k is written as

$$rk_{i\to k} := (\beta_{iku}, \beta_{iku} \cdot s_k + pe_{iku} - s_i \cdot (2^r)^u)_{u\in\{0,1,\ldots\lfloor \log_2(q)/r\rfloor\}},$$

  where β_{iku}, e_{iku} are generated by party k.
- The challenge ciphertext related to party i^* is $c^* = (c_0^*, c_1^*) \in R_q^2$:

$$c_0^* := b^* \cdot v^* + pe_0^* + m_b \in R_q,\ c_1^* := a^* \cdot v^* + pe_1^* \in R_q,$$

  where $b \in \{0, 1\}$ is the challenge bit, $v^*, e_0^*, e_1^* \leftarrow \chi_e$, and (a^*, b^*) is the challenge public key.

Let Game_k, $i \in \{1 \le k \le N\}$, be defined by considering the honest party $k \in \Gamma_H$. Game_k is identical to Game_{k-1} except for the following changes:

- When generating the k-th key pair, b_k is a randomly generated ring element rather than a RLWE sample.
- When answering the re-encryption key query (i, k): First, note that $i > k$ because of the topological ordering. The re-encryption key is expressed as

$$rk_{i\to k} := (\beta_{iku}, \gamma_{iku})_{u\in\{0,1,\ldots\lfloor \log_2(q)/r\rfloor\}},$$

  where γ_{iku} is freshly random.

Each Game_k is computationally indistinguishable from Game_{k-1} because of the RLWE assumption. First, $k \in \Gamma_H$ and therefore, there is no re-encryption "edge" from user k to any user in Γ_C. Additionally, as before, all the re-encryption keys (k, i) have already been replaced with uniformly random ring elements in the prior games. Consequently, the secret key s_k is used only in the form of fresh RLWE samples in its public key and in the re-encryption keys. Thus, all these can be replaced by uniformly random ring elements by invoking the RLWE assumption. The security loss is proportional to the number of re-encryption key and re-encryption queries that user k was part of (an additional multiplicative factor $1 + \lfloor \log_2 q/r \rfloor$ is incurred in the security loss as each re-encryption key contains that many RLWE samples).

$\text{Game}_{\text{final}}$ is same as Game_N except for the challenge ciphertext that is expressed as

$$c_0^* := r_1^* + m_b \in R_q,\ c_1^* := r_2^* \in R_q,$$

where r_1^*, r_2^* are freshly random ring elements in R_q. This is computationally indistinguishable from Game_N by the RLWE assumption as well.

The last change guarantees that the challenge bit b is information-theoretically hidden from $\mathcal{A}$, and therefore, the advantage of the adversary in Game_1 is 0.

Putting together, we see that

$$\mathbf{Adv}_{\mathcal{A}}^{cpa}(\lambda) \le (\rho \cdot (Q_{rk} + Q_{re}) + N + 1) \cdot \mathbf{Adv}_{\mathcal{D}}^{RLWE_{\phi,q,\chi_e}}(\lambda)$$

where $\rho := 1 + \lfloor \log_2 q/r \rfloor$. This finishes our proof. □

6 Parameter Selection

A general issue with lattice encryption schemes is that they are more complicated to parameterize than other families of encryption schemes. Parameter selection is governed largely by a correctness condition (which is specific to the scheme being analyzed) and security conditions for the underlying security assumptions.

For NTRU-ABD-PRE, parameter selection is governed by the correctness condition (6), the security condition (3) accounting for the NTRU immunity against subfield lattice attacks, and RLWE security condition (4).

For BV-PRE, parameter selection is governed by the correctness condition (8) and RLWE security condition (4).

We identify the parameter tradeoffs associated with the correctness constraint and security constraints for both schemes in the experimental results section of this paper. Of high importance are the ring dimension n and ciphertext modulues q which have the largest direct impact on the runtimes of the scheme. The value of n should be kept as small as possible as runtime is at least linear in n for all operations. The value of q determines the sizes of integers that need to be manipulated computationally. Ideally q should be kept less than the threshold 2^{32} or the less preferred threshold 2^{64} so that operations can be supported by native arithmetic operations supported with the processor word sizes in modern processors.

The value of d, the number of hops that the re-encryption scheme supports, can be thought of as an application-specific parameter determined by the number of PRE hops needed.

We begin the process of parameter selection with the security parameter δ, also known as the root Hermite factor. The root Hermite factor is discussed above in the Related Work section with relevant references. A heuristic argument is presented in [12] which suggests that a root Hermite factor of $\delta = 1.006$ could provide adequate security. Therefore, we select to be as close as possible to the ceiling $\delta < 1.006$.

The bound for discrete Gaussian distribution $\chi_i(x)$, where $i \in (k, e)$, is expressed as $B_i = \sigma_i\sqrt{\alpha}$, where σ_i is the standard deviation of the distribution and α determines the effective probability that a coefficient generated using discrete Gaussian distribution (or a product of discrete Gaussians) exceeds the bound B_i [23].

The value of σ_e is usually chosen in the range from 3 to 6, and we set the value of σ_e to 4 as in [17]. We set α to 9, which for the case of an integer generated using discrete Gaussian distribution corresponds to the theoretic probability of at most 2^{-8} of choosing a value that exceeds the upper bound B_i.

We validated our selection of σ_i and α experimentally. Over 35,000 iterations of encryption/decryption (using different keys) for ring dimensions in the range from 2^9 to 2^{15} (5,000 iterations for each value of ring dimension), we observed no decryption errors. Note that when products of two discrete Gaussians (encryption scheme), three discrete Gaussians (single-hop re-encryption in the case of NTRU-ABD-PRE), and higher number of discrete Gaussians (multi-hop re-encryption in the case of NTRU-ABD-PRE) are considered, the practical probability drops dramatically. This implies that smaller practical values of α may be possible.

Subsequent to the selections of d, δ, σ_e, and α, we can choose n, q, and σ_k (only in the case of NTRU-ABD-PRE) experimentally using appropriate correctness and security constraints to minimize runtime/throughput for various values of the relinearization window r and plaintext modulus p.

Table 3 shows the minimum number of bits needed to represent the ciphertext modulus q (which

Table 3: Dependence of minimum bits required to represent modulus q for selections of ring dimension n and multiple re-encryption depths d at $p = 2$ and $r = 1$.

PRE Scheme	d	**Ring dimension** n					
		512	1024	2048	4096	8192	16384
NTRU-ABD-PRE	1	–	35	36	37	38	39
	2	–	–	–	93	96	99
	3	–	–	–	–	–	–
BV-PRE	1	17	18	18	19	19	20
	2	18	18	19	19	20	20
	3	18	19	19	20	20	21

Table 4: Dependence of minimum values of ring dimension n and the number of bits k required to represent the ciphertext modulus q, on plaintext modulus p and relinearization window r for re-encryption depth d of unity.

PRE Scheme	p	$r = 1$		$r = 2$		$r = 4$		$r = 8$		$r = 16$	
		n	k	n	k	n	k	n	k	n	k
NTRU-ABD-PRE	2	1024	35	1024	35	1024	35	1024	38	2048	48
	16	2048	53	2048	53	2048	53	2048	53	2048	57
	256	4096	78	4096	78	4096	78	4096	78	4096	78
	4096	4096	102	4096	102	4096	102	4096	102	4096	102
	65536	4096	126	4096	126	4096	126	4096	126	4096	126
BV-PRE	2	512	17	512	17	512	18	1024	22	1024	29
	16	512	20	1024	21	1024	22	1024	25	1024	32
	256	1024	25	1024	25	1024	26	1024	29	1024	37
	4096	1024	29	1024	29	1024	30	1024	33	2048	41
	65536	1024	33	1024	33	1024	35	1024	37	2048	45

we refer to as $k = \lfloor \log_2 q + 1 \rfloor$), as a function of ring dimension n and re-encryption depth d for the relinearization window of unity assuming the other parameters were selected as above. It can be seen that NTRU-ABD-PRE requires a ring dimension of at least 4096 and ciphertext modulus of approximately 100 bits to support two re-encryption hops in contrast to a ring dimension of 512 and 18-bit ciphertext modulus for BV-PRE, implying that NTRU-ABD-PRE can be treated as a single-hop scheme for all practical purposes. It should also be noted that all ciphertext moduli for BV-PRE require representations with less than 32 bits, thus enabling efficient implementations based on native integer types (32-bit and 64-bit integers). The growth of the number of ciphertext modulus bits k with increase in ring dimension n for both schemes can be easily estimated from expressions (6) and (8): for NTRU-ABD-PRE the dependence of q on n is $n^{1+d/2}$ while for BV-PRE, it is $\sqrt{n}$.

Table 4 illustrates the effect of increasing the relinearization window r and plaintext modulus p on the minimum values of ring dimension n and the number of bits k required to represent the ciphertext modulus q for both schemes. Increase in r reduces the dimension of the re-encryption

Table 5: Dependence of minimum values of ring dimension n and the number of bits k required to represent the ciphertext modulus q, on re-encryption depth d and relinearization window r for BV-PRE at $p = 2$.

d	$r = 1$		$r = 2$		$r = 4$		$r = 8$		$r = 16$	
	n	k	n	k	n	k	n	k	n	k
1	512	17	512	17	512	18	1024	22	1024	29
2	512	18	512	18	512	19	1024	23	1024	30
5	512	19	512	19	512	20	1024	24	1024	31
10	512	19	512	20	1024	22	1024	25	1024	32
20	512	20	1024	21	1024	23	1024	25	1024	33
50	1024	22	1024	23	1024	24	1024	28	1024	35
100	1024	23	1024	24	1024	25	1024	29	1024	36

key (to $\lfloor \log(q)/r \rfloor + 1$) and the number of NTT operations (performed for groups of r bits of each coefficient), which effectively reduces the re-encryption runtime. Increase in p improves the plaintext throughput of PRE and reduces the ciphertext expansion factor defined as $k/\lfloor \log_2 p + 1 \rfloor$. More detailed information on these performance metrics is presented in Section 8.

The results in Table 4 suggest that r can be used to reduce the re-encryption runtime with negligible effect on the encryption/decryption runtimes. For instance, the ring dimension and the number of bits k required to represent the ciphertext modulus q are essentially the same for BV-PRE at $p = 2$ when r is increased from 1 to 4. At the same time, this reduces the runtime of re-encryption by roughly a factor of 4. One can also observe for BV-PRE that increasing the plaintext modulus to 65536 (2 bytes per polynomial coefficient) raises the ring dimension requirement only by a factor of 2 (to 1024), which implies that a much higher plaintext throughput can be achieved for BV-PRE by using large values of plaintext modulus (in applications where runtime/latency is not critical). It can also be seen that the ring dimension / ciphertext modulus requirements are substantially lower for BV-PRE as compared to NTRU-ABD-PRE.

Table 5 shows the effect of increasing the re-encryption depth d for different values of relinerization window r on the minimum values of ring dimension n and the number of bits k required to represent the ciphertext modulus q for BV-PRE (results for NTRU-ABD-PRE are not presented because the values of ring dimension and ciphertext modulus are impractical for $d = 2$). It can be seen that BV-PRE supports at least 20 re-encryption hops at $n = 512$ (the maximum number is 23 hops). One can also observe that the number of bits k required to represent the ciphertext modulus q changes slowly with increase in re-encryption depth because the noise growth is additive (rather than multiplicative as in the case of NTRU-ABD-PRE), and 100 re-encryption hops can be supported without exceeding the ring dimension of 1024.

7 Software Implementation

7.1 Software Library Design

We implemented our PRE scheme in PALISADE, a general-purpose portable multi-threaded C++ library designed to support and ease the development of lattice-based encryption prototypes.

The main runtime performance bottleneck is conversion between coefficient and evaluation representations. For the power-of-two cyclotomic rings, the most efficient algorithm to perform this operation is the Fermat-Theoretic Transform (FTT) [4]. We implemented FTT with NTT as a subroutine in PALISADE. For NTT, the iterative Cooley-Tukey algorithm with optimized butterfly operations was applied. The two slowest sub-operations needed to support NTT operations are multiplication and modulo reduction. For multiplication, we used the standard shift-and-add multiplication algorithm as it performs well for relatively small ciphertext moduli (up to multiple hundreds of bits, but in our case the running bitwidths required to represent ciphertext moduli do not exceed 128 bits). For modulo reduction, we used the generalized Barrett modulo reduction algorithm [14], which requires one pre-computation per NTT run and converts one modulo reduction to roughly two multiplications. For discrete Gaussian sampling, we used the inversion method from [32].

The conventional relinearization procedure works with the relinearization window r of unity, implying that every coefficient of the ciphertext polynomial c is decomposed into bits. Although this technique dramatically reduces the noise growth (from $\|c\|_\infty$ to 1), it significantly increases both computational and space complexities of re-encryption. As there is no efficient method to extract bits from a polynomial in CRT form, the ciphertext polynomial c has to be first converted to the coefficient form, then decomposed into polynomials over $\mathbb{Z}_2$, and finally all of these bit-level polynomials need to be converted back to CRT form prior to performing the component-wise multiplication with the elements of the re-encryption key. The total computational cost of this operation is $\lfloor \log_2(q) \rfloor + 2$ FTT operations. The size of the re-encryption key is approximately $n \cdot \log_2(q)^2$ bits (or the double of that in the case of BV-PRE).

To reduce the number of FTT operations and size of the re-encryption key, we consider a generalized relinearization window of up to 16 bits. It can be seen that in the case of $r = 8$, the number of FTT operations reduces to $\lfloor \log_2(q)/8 \rfloor + 2$ and the re-encryption key size reduces by a factor of 8. At the same time, Table 4 suggests that the number of bits required to represent q, which is determined by the correctness constraint, increases only by 3 bits compared to the case of $r = 1$ with the minimum ring dimension n staying at the same level (for NTRU-ABD-PRE at $p = 2$). In view of the above, it can be expected that the re-encryption time for this case will be significantly less than for $r = 1$, which is demonstrated in the next section.

8 Experimental Evaluation

8.1 Methodology

We identify a set of standard metrics, including those used in related work [3, 31] with which to evaluate the performance of our PRE design and implementation. These metrics include:

1. Runtime / Latency: How long it takes to perform the implemented Encryption, Re-Encryption and Decryption operations for various parameter settings.

2. Throughput: How many plaintext bits per unit time can be processed by the implemented operations for various settings.

3. Ciphertext Expansion: How many bits are required to represent ciphertext for every bit in the plaintext.

4. Memory Usage: How much memory is required to run the implemented operations for various settings.

We would normally also use security as a metric to evaluate the performance of our PRE design and implementation, but we assume a ceiling on the security parameter such that $\delta < 1.006$, and we would want δ to be as close as possible to 1.006 to provide as quick runtime performance as possible while providing adequate security. For our experimental analyses, we varied the ring dimension n, relinearization window r, plaintext modulus p, and number of hops supported d to explore tradeoffs in runtime and amortized throughput.

Because we perform all experiments in the single-threaded mode and our implementation does not access disk or networking interfaces, we use latency as a means of determining the temporal overhead of the implementation. Further, runtime performance is useful, for example, when assessing fitness for real-time applications when end-to-end latency is critical. We use the throughput metric to assess how much plaintext data can be processed by the implementation per unit time.

Related to ciphertext expansion is memory usage. Memory intensive operations may not be easily supported on resource-constrained devices, such as embedded systems used for disposable sensor nodes. We therefore differentiate between the memory requirements of PRE clients (subscribers and publishers) from those of PRE servers (brokers). Memory usage for PRE clients is governed primarily by the size of public/private keys and ciphertext elements. At the same time, the memory requirements for PRE servers are determined primarily by the size of re-encryption keys and decomposed ciphertext ring elements.

We conducted experiments for our PRE implementation on a commodity desktop computing environment. The evaluation environment used an Intel Core i7-3770 CPU rated at 3.40GHz and 16GB of memory running CentOS 7. All of our implementations were compiled as single-threaded and used only one core despite our test environment providing multiple cores.

We generated random plaintext samples using discrete uniform distribution from 0 to $p - 1$. We ran 100 iterations for a subset of parameter datasets listed in Tables 3-5 and evaluated the mean runtime of encryption, decryption, and re-encryption operations, with decryption runtime measured before and after re-encryption, and the runtime of multiple re-encryptions. The number of correct decryptions was also recorded, and no decryption errors were observed.

In Tables 7 through 9 we present experimental results for the dependence of runtime and throughputs of PRE operations on variations in key configuration parameters, including the ring dimension, relinearization window, plaintext modulus, and number of re-encryption hops. We show throughputs in kilobits per second (Kbps) for encryption, re-encryption, and decryption amortized in terms of the plaintext size.

The ciphertext expansion factor is equal to $k = \lfloor \log_2 q + 1 \rfloor$ in all tables except for 8. Although not directly related to the security provided, the key size in bits is equal to the ring dimension n times the number of bits k in ciphertext modulus q.

We consider key generation to be an offline process which is run once for most feasible applications of the PRE capability. For all of our experimental configurations we observed key generation and proxy key generation runtime of less than 1 second.

8.2 Single-Hop Re-Encryption

Table 6 shows the effect of changes in ring dimension n on runtime, amortized throughputs, and ciphertext expansion factors for single-hop re-encryption using both NTRU-ABD-PRE and BV-

Table 6: Experimental runtime performance of encryption, decryption, and re-encryption operations for ring dimension n at r=1, p=2, and d=1.

Configuration			Runtime				Throughput		
PRE Scheme	n	k	**Enc (ms)**	**Dec before ReEnc (ms)**	**ReEnc (ms)**	**Dec after ReEnc (ms)**	**Enc (Kbps)**	**ReEnc (Kbps)**	**Dec after ReEnc (Kbps)**
NTRU-ABD-PRE	1024	35	2.13	2.45	67.08	2.44	481.06	15.26	418.85
	2048	36	4.62	5.27	150.95	5.26	443.27	13.57	389.71
	4096	37	9.80	11.07	331.73	11.05	417.94	12.35	370.78
	8192	38	20.69	23.35	724.80	23.29	395.95	11.30	351.70
	16384	39	44.15	49.73	1597.81	49.41	371.14	10.25	331.58
BV-PRE	512	17	0.85	0.76	11.77	0.76	604.37	43.51	674.62
	1024	18	1.81	1.64	27.48	1.63	567.03	37.26	628.04
	2048	18	3.84	3.47	59.83	3.44	533.82	34.23	594.85
	4096	19	7.99	7.24	131.68	7.22	512.70	31.11	566.95
	8192	19	17.00	15.80	296.63	15.33	481.85	27.62	534.30
	16384	20	35.77	32.82	634.71	32.70	458.07	25.81	501.10

PRE schemes with security parameter $\delta \leq 1.006$. The highest encryption, re-encryption, and decryption throughputs and lowest runtime are observed for the smallest ring dimension: 1024 and 512 for NTRU-ABD-PRE and BV-PRE, respectively. The ciphertext expansion, which is proportional to the number of bits k required to represent the ciphertext modulus q, and memory usage for both PRE clients and servers, which is proportional to the product of ring dimension and the number of bits k required to represent the ciphertext modulus q, are lowest for the smallest value of ring dimension. This implies that one should always choose the smallest ring dimension satisfying the desired security level.

Note that the runtimes for BV-PRE scheme operations are always lower than for NTRU-ABD-PRE due to lower requirements on the ring dimension and the number of bits required to represent the ciphertext modulus of the former. For the same ring dimension, the runtime improvement factors observed from Table 6 are approximately 1.2 for encryption, 1.5 for decryption, and 2.5 for re-encryption operations. Considering that the lowest ring dimension with security parameter $\delta \leq 1.006$ for BV-PRE is 512, the improvement factors for throughputs at smallest ring dimension are 1.3, 1.6, and 2.9 for encryption, decryption, and re-encryption, respectively. The decryption times after regular encryption and proxy re-encryption are approximately the same for all datasets, which also applies to all other experimental results presented in this paper.

Table 7 shows the dependence of runtime and throughputs on variations in the relinearization window r for single-hop re-encryption with security parameter $\delta \leq 1.006$. The plaintext modulus is kept the same ($p = 2$). It can be seen that for both schemes the highest encryption runtime and throughput are observed for $r = 1$ (which uses the smallest the number of bits required to represent the ciphertext modulus.) As the relinearization window r increases, the re-encryption time declines until the ring dimension is forced to double by the security constraint. This lowest re-encryption runtime occurs at $r = 8$ and $r = 4$ for NTRU-ABD-PRE and BV-PRE, respectively. Note that the

Table 7: Experimental runtime performance of encryption, decryption, and re-encryption operations on relinearization window size r at p=2 and d=1.

Configuration				Runtime				Throughput		
PRE Scheme	r	n	k	**Enc (ms)**	**Dec before ReEnc (ms)**	**ReEnc (ms)**	**Dec after ReEnc (ms)**	**Enc (Kbps)**	**ReEnc (Kbps)**	**Dec after ReEnc (Kbps)**
NTRU-ABD-PRE	1	1024	35	2.13	2.45	67.08	2.44	481.06	15.26	418.85
	2	1024	35	2.13	2.45	36.65	2.44	480.75	27.94	419.03
	4	1024	35	2.13	2.45	21.27	2.44	480.09	48.15	418.86
	8	1024	38	2.11	2.46	13.88	2.44	484.61	73.77	418.95
	16	2048	48	4.72	5.45	19.97	5.42	434.23	102.57	377.71
BV-PRE	1	512	17	0.85	0.76	11.77	0.76	604.37	43.51	674.62
	2	512	17	0.83	0.74	6.38	0.74	615.12	80.21	691.43
	4	512	18	0.84	0.77	4.33	0.76	607.58	118.27	676.02
	8	1024	22	1.78	1.60	6.23	1.60	576.85	164.25	639.50
	16	1024	29	2.00	1.82	5.41	1.82	512.23	189.15	562.60

encryption and decryption runtimes for these values of r are approximately the same as for $r = 1$, which implies that $r = 8$ and $r = 4$ are optimal values for all operations of NTRU-ABD-PRE and BV-PRE, respectively, from the runtime/latency perspective. At the same time, the re-encryption throughput at $r = 16$ is highest for both schemes. This implies that in applications where re-encryption througput needs to be maximized and latency requirements are low, $r = 16$ could be the preferred choice. It should be noted that the ciphertext expansion grows with r, memory usage by PRE clients increases proportionally to the number of bits required to represent the ciphertext modulus and ring dimension, and memory usage by PRE servers declines as the re-encryption keys are composed of $\lfloor \log_2 (q) / r \rfloor + 1$ ring elements.

Table 8 illustrates the effect of plaintext modulus p on performance metrics of PRE operations for both schemes. The relinearization window is kept constant ($r = 1$). One can see that runtime increases as p rises due to increased requirements on the number of bits k required to represent the ciphertext modulus q and the ring dimension n. At the same time, plaintext throughputs increase until $p = 4096$ for both schemes. Ciphertext expansion factors, defined as $k/\log_2 p$, are highest at $p = 65536$. This suggests that larger plaintext moduli may be suggested when high throughput and low ciphertext expansion are sought, and latency requirements are secondary. One can also see that the memory usage of both PRE clients and servers increases with p due to requiring more bits to represent the ciphertext modulus and larger ring dimensions, which may be an issue for embedded systems (PRE clients).

All tables for single-hop re-encryption suggest that BV-PRE outperforms NTRU-ABD-PRE for all performance metrics. The best BV-PRE re-encryption runtime of 4.33 ms is almost two orders of magnitudes faster than the runtime reported for comparable conditions (same ring dimension of 512) in the independent work of [31]. Besides being faster than the scheme reported in [31], BV-PRE is unidirectional and is based strictly on the RLWE assumption.

We observed experimentally that as ring dimension n increases for our schemes, the latency due

Table 8: Experimental runtime performance of encryption, decryption, and re-encryption operations on plaintext modulus p at r=1 and d=1.

Configuration				Runtime				Throughput		
PRE Scheme	p	n	k	**Enc (ms)**	**Dec before ReEnc (ms)**	**ReEnc (ms)**	**Dec after ReEnc (ms)**	**Enc (Kbps)**	**ReEnc (Kbps)**	**Dec after ReEnc (Kbps)**
NTRU-ABD-PRE	2	1024	35	2.13	2.45	67.08	2.44	481.06	15.26	418.85
	16	2048	53	5.38	7.73	228.30	7.55	1523.62	35.88	1085.06
	256	4096	78	16.20	23.05	1016.58	22.98	2022.23	32.23	1425.92
	4096	4096	102	20.00	28.94	1642.66	28.88	2458.12	29.92	1701.95
	65536	4096	126	21.24	33.66	2141.11	34.03	3085.50	30.61	1925.83
BV-PRE	2	512	17	0.85	0.76	11.77	0.76	604.37	43.51	674.62
	16	512	20	0.95	0.91	13.84	0.92	2156.82	147.93	2221.67
	256	1024	25	2.03	1.90	36.65	1.95	4028.96	223.53	4200.40
	4096	1024	29	2.33	2.15	47.28	2.19	5269.21	259.90	5600.50
	65536	1024	33	2.74	2.47	63.57	2.41	5989.05	257.73	6809.95

to Encryption, Re-Encryption and Decryption increase, but the ammortized cost and throughput decrease. Furthermore, ciphertext expansion and memory requirements increase. The effect of increasing the relinearization window r is similar to the effects of increasing n, except that Re-Encryption latency decreases and throughput increases. The effects of increasing plaintext modulus p is similar to the effect of increasing ring dimension. These results may be used for selecting optimal configuration of these three parameters in practical single-hop PRE applications.

8.3 Multi-Hop Re-Encryption

Table 9 illustrates the dependence of runtime, throughputs, and ciphertext expansion factors on the number of re-encryption hops for PRE-BV with security parameter $\delta \leq 1.006$. The results for NTRU-ABD-PRE are not listed because the scheme supports only two re-encryption hops with the second hop requiring more bits k to represent the ciphertext modulus q, as seen in Table 3. It can be seen that PRE-BV scales well with re-encryption depth. For the first 20 hops, the runtime and throughput metrics are approximately the same for encryption and decryption operations, and degrade by at most 20% for the re-encryption operation. For larger re-encryption depths (up to 100), the encryption/decryption throughputs degrade only by at mosty 20% as compared to the single-hop case while re-encryption throughput declines by 40%. It should be noted that the observed enryption/decryption times are still under 2 ms, which may be adequate for many practical applications.

The runtimes for first re-encryption hop and last re-encryption hop are essentially the same for all re-encryption depths, with the latter being slightly lower due to local caching effects of the implementation. The decryption times after regular encryption and proxy re-encryption are approximately the same.

Table 9: Dependence of performance metrics for BV-PRE encryption, decryption, and re-encryption operations on the number of re-encryption hops d at r=1 and p=2.

Configuration			Runtime					Throughput		
d	n	k	**Enc (ms)**	**Dec before ReEnc (ms)**	**First ReEnc (ms)**	**Last ReEnc (ms)**	**Dec after ReEnc (ms)**	**Enc (Kbps)**	**ReEnc (Kbps)**	**Dec after last ReEnc (Kbps)**
1	512	17	0.85	0.76	11.77	–	0.76	604.37	43.51	674.62
2	512	18	0.86	0.77	12.84	12.82	0.77	597.52	39.88	667.48
5	512	19	0.85	0.76	13.74	13.43	0.76	604.31	37.27	672.78
10	512	19	0.85	0.77	13.56	13.57	0.76	602.13	37.75	670.37
20	512	20	0.84	0.76	13.69	13.69	0.75	611.51	37.40	681.35
50	1024	22	1.83	1.67	33.98	33.60	1.67	560.44	30.13	613.60
100	1024	23	2.00	1.87	39.45	39.38	1.86	512.11	25.96	550.19

9 Application

A major security challenge for Pub/Sub systems is confidentiality of information which is distributed by the Pub/Sub broker. Existing Pub/Sub systems protect information payloads via encryption that requires either: 1) the publisher and subscriber coordinate to establish the encryption and decryption keys or 2) the Pub/Sub broker decrypts the information payloads from the publishers and then encrypts this information payload again for re-transmission to the subscribers. The first solution contradicts one of the goals of Pub/Sub systems, i.e., the decoupling of publishers and subscribers. The second solution solves this issue, but gives the broker access to the unprotected information. Thus, it makes the broker a ripe target for adversaries to compromise and steal sensitive information.

PRE is a natural fit to support publish-subscribe because PRE maintains data confidentiality even when the broker is compromised and an adversary obtains all re-encryption keys and observes all communications between the publisher, broker and subscriber. These features reduce the need for special, difficult to use security-enabled hardware and software for high-assurance applications, such as in military settings. A compromised PRE-enabled broker would at most allow the adversary to learn which subscribers are allowed to receive information from which publishers based on the existence of re-encryption keys.

In this section we are particularly interested in understanding how to parameterize the PRE schemes for three application use cases to illustrate the adaptability of our design and implementation.

9.1 Enterprise Security

PRE could be very useful in enterprise-style computing environments such as for medical file sharing. Enterprise environments are characterized by high resource availability - both computational power at the publishers, subscribers and PRE servers, but also bandwidth availability. The primary concern would be overall throughput.

For single-hop applications, the goal is to maximize re-encryption throughput. As tables 7 and 8 suggest, re-encryption throughput can be maximized by increasing the relinearizaton window r or increasing the plaintext modulus p (up to certain limits, until the ciphertext modulus bit length and ring dimension start to significantly slow down the runtime). In the case of the BV-PRE scheme, the plaintext throughputs can reach 250 Kbps. The combined effect of increased plaintext modulus and relinearization window can produce even higher plaintext outputs but this analysis should be performed based on the requirements of a specific application. The BV-PRE scheme can also provide a multi-hop capability without significantly increasing parameter requirements if the value of relinearization window r in expression (8) does not exceed 8.

9.2 Embedded Support

At the opposite end of the resource availability spectrum is the use case of embedded sensors that collect, encrypt and publish data to a PRE server. To set up the environment, point-to-point communication approvals need to be established, namely that:

- The sensors would need to have appropriate encryption keys.
- The sensors would need to be paired with the PRE server.
- The approval for subscribers to receive data would need to be received to approve the generation of a re-encryption key hosted at the PRE server.

PRE addresses the above measures to encrypt data at the sensor, transmit the data to a cloud storage environment where processing is done, and the encrypted results shared with intended recipients, without ever decrypting the data or sharing decryption keys. Recent results [22, 10] show that it is possible to implement public key lattice encryption schemes, very similar to our PRE schemes, and run them on very resource-limited devices, inlcuding devices using 8-bit AVR processors [22]. These results also provide general design guidelines to port our designs into limited hardware.

Because embedded use cases require computationally intense operations at low-powered sensor nodes, encryption throughput is paramount. It is feasible that multi-hop encryption would be needed so that the encrypted information can aggregate from the sensors to local PRE servers which send data to a centralized encrypted information clearinghouse. In this situation, the use of BV-PRE with $r = 1$ and a large plaintext modulus, for example, $p = 65536$, would maximize encryption throughput.

An alternative formulation of this use-case for especially low-powered sensor devices might rely on processors with 32-bit words, or less. In this scenario it is generally important for modulus bit-widths to be within a power-of-2 rather than without for increased performance. If the modulus bit-width is larger than bit-width of the processor, then extra shuffling of data and at least a factor-of-2 decrease in performance is likely to result. Selecting BV-PRE at $n = 512$, $r \in (1, 2, 4)$, and a ciphertext modulus bit-width of 17-18 bits is recommended to maximize encryption throughput. It should be noted that the ciphertext bit-width of up to 20 and ring dimension of 512 can support up to 23 re-encryption hops of BV-PRE at $p = 2$ and $r = 1$, as can be seen from Tables 5 and 9.

9.3 Hybrid Deployment with AES

This work is motivated by the problem of sharing data across coalition partners who do not interact directly, including across administrative boundaries, yet want to control data access within each coalition partner by policy. While encryption and policy enforcement solutions are available, a major challenge is the lack of suitable techniques to generate or share encryption keys. For example, streaming video, images and text data are often transmitted when encrypted by AES, because AES is considered both secure and highly efficient. PRE can be used in these scenarios as an AES key distribution mechanism.

Single-hop application operation of PRE would provide the most control for users to limit the spread of restricted data. Based on the RLWE security constraint and PRE correctness constraint, we should keep q as small as possible to guarantee correctness and use the lowest value of n that satisfies the security requirements.

10 Discussion and Ongoing Activities

In this paper we present two new lattice-based PRE schemes. We experimentally evaluate the performance of the PRE schemes. Our lattice encryption library is an important aspect of our implementation performance in that its modularity and extensibility enables us to further improve performance with either improved mathematical libraries or even hardware acceleration as these technologies become available.

A benefit of our PRE approach is that it supports applications on commodity computing hardware and improves the overall security of information sharing in practical pub/sub systems. Taken together, this could greatly reduce the operational costs of highly regulated industries such as health-care where regulatory compliance restricts the ability to outsource computation to low cost cloud computing environments.

Although we have focused our discussion on PRE for situations with one producer (Alice) and one consumer (Bob), there is no theoretical limit to the number of producers and subscribers that can be used for PRE operations. With PRE we can support general many-to-many operations where data from many producers is securely shared with many consumers through the PRE prototype, by generating multiple re-encryption keys, one for every permitted publisher-subscriber information sharing pair. A possible approach to address scalability is to distribute the operations of the PRE servers across many computation nodes, and we seek to address this in follow-on research.

References

[1] M. Albrecht, S. Bai, and L. Ducas. *A Subfield Lattice Attack on Overstretched NTRU Assumptions*, pages 153–178. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[2] Y. Aono, X. Boyen, L. T. Phong, and L. Wang. Key-private proxy re-encryption under LWE. In *Progress in Cryptology–INDOCRYPT 2013*, pages 1–18. Springer, 2013.

[3] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30, 2006.

[4] A. Aysu, C. Patterson, and P. Schaumont. Low-cost and area-efficient fpga implementations of lattice-based cryptography. In *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on*, pages 81–86, June 2013.

[5] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In *Advances in CryptologyEUROCRYPT'98*, pages 127–144. Springer, 1998.

[6] J. Bos, K. Lauter, J. Loftus, and M. Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In M. Stam, editor, *Cryptography and Coding*, volume 8308 of *Lecture Notes in Computer Science*, pages 45–64. Springer Berlin Heidelberg, 2013.

[7] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13, 2014.

[8] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 97–106, Washington, DC, USA, 2011. IEEE Computer Society.

[9] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, pages 505–524, 2011.

[10] J. Buchmann, F. Göpfert, T. Güneysu, T. Oder, and T. Pöppelmann. High-performance and lightweight lattice-based public-key encryption. In *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, pages 2–9. ACM, 2016.

[11] R. Canetti and S. Hohenberger. Chosen-ciphertext secure proxy re-encryption. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 185–194. ACM, 2007.

[12] Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011.

[13] J. H. Cheon, J. Jeong, and C. Lee. An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low-level encoding of zero. *LMS Journal of Computation and Mathematics*, 19:255–266, 1 2016.

[14] J.-F. Dhem and J.-J. Quisquater. Recent results on modular multiplications for smart cards. In J.-J. Quisquater and B. Schneier, editors, *Smart Card Research and Applications*, volume 1820 of *Lecture Notes in Computer Science*, pages 336–352. Springer Berlin Heidelberg, 2000.

[15] J. Fan and F. Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. `http://eprint.iacr.org/2012/144`.

[16] X. Fan and F.-H. Liu. Various proxy re-encryption schemes from lattices. Cryptology ePrint Archive, Report 2016/278, 2016. `http://eprint.iacr.org/2016/278`.

[17] C. Gentry, S. Halevi, and N. Smart. Homomorphic evaluation of the AES circuit. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 850–867. Springer Berlin / Heidelberg, 2012.

[18] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In J. P. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer Berlin Heidelberg, 1998.

[19] A.-A. Ivan and Y. Dodis. Proxy cryptography revisited. In *NDSS*, 2003.

[20] E. Kirshanova. Proxy re-encryption from lattices. In *Public-Key Cryptography–PKC 2014*, pages 77–94. Springer, 2014.

[21] R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In *CT-RSA*, pages 319–339, 2011.

[22] Z. Liu, H. Seo, S. Sinha Roy, J. Großschädl, H. Kim, and I. Verbauwhede. *Efficient Ring-LWE Encryption on 8-Bit AVR Processors*, pages 663–682. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[23] A. López-Alt, E. Tromer, and V. Vaikuntanathan. Multikey fully homomorphic encryption and on-the-fly multiparty computation. *IACR Cryptology ePrint Archive*, 2013:94, 2013. Full Version of the STOC 2012 paper with the same title.

[24] V. Lyubashevsky, C. Peikert, and O. Regev. *On Ideal Lattices and Learning with Errors over Rings*, pages 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[25] V. Lyubashevsky, C. Peikert, and O. Regev. *A Toolkit for Ring-LWE Cryptography*, pages 35–54. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[26] D. Micciancio. Duality in lattice cryptography. In *Public Key Cryptography*, 2010. Invited talk.

[27] D. Micciancio. Lattice-based cryptography. In *Encyclopedia of Cryptography and Security*, pages 713–715. Springer, 2011.

[28] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

[29] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004.

[30] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post Quantum Cryptography*, pages 147–191. Springer, February 2009.

[31] D. Nuñez, I. Agudo, and J. Lopez. NTRUReEncrypt: An efficient proxy re-encryption scheme based on NTRU. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 179–189. ACM, 2015.

[32] C. Peikert. An efficient and parallel gaussian sampler for lattices. In T. Rabin, editor, *Advances in Cryptology CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer Berlin Heidelberg, 2010.

[33] L. T. Phong, L. Wang, Y. Aono, M. H. Nguyen, and X. Boyen. Proxy re-encryption schemes with key privacy from lwe. Cryptology ePrint Archive, Report 2016/327, 2016. `http://eprint.iacr.org/2016/327`.

[34] O. Regev. Quantum computation and lattice problems. *SIAM J. Comput.*, 33(3):738–760, 2004. Preliminary version in FOCS 2002.

[35] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. G. Paterson, editor, *Advances in Cryptology EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 27–47. Springer Berlin Heidelberg, 2011.

[36] J. van de Pol. Quantifying the security of lattice-based cryptosystems in practice. In *Mathematical and Statistical Aspects of Cryptography*, 2012.