# Improvement of Lattice-Based Cryptography Using CRT

Thomas Plantard, Mike Rose, and Willy Susilo

School of Computer Science and Software Engineering
University of Wollongong, Wollongong NSW, Australia
{thomaspl,mrose,wsusilo}@uow.edu.au

**Abstract.** In this paper, we first critically analyze two existing lattice-based cryptosystems, namely GGH and Micciancio, and identify their drawbacks. Then, we introduce a method for improving the implementation of GGH using the Chinese Remainder Theorem (CRT). Furthermore, we also propose another cryptosystem optimized for CRT, drawing on the strengths of both cryptosystems. We provide a fair comparison between our scheme and the existing ones.

## 1 Introduction and Motivation

With the continuous advancements in the field of quantum computing, the security of many existing asymmetric key cryptosystems has been demonstrated to be broken in the theoretical sense, with computational insecurity of these schemes being dependent only on technological advancements. As a result, new one-way trapdoor functions must be developed that will remain secure after quantum computers become available. One promising avenue of research in this direction is lattice-based cryptography.

Recent advancements in the field of lattice-based cryptography have brought a sustained interest in producing a lattice-based cryptosystem that runs in a similar space and time complexity as existing conventional asymmetric key cryptosystems. Two specific related cryptosystems showing much promise are the cryptosystems introduced by Goldreich, Goldwasser and Halevi [5] (GGH), and its modification and improvement by Micciancio [9].

While the security of any cryptosystem in an information theoretic sense relies on the space and time complexity of the 'trapdoor function' being used, we need a stronger definition to have the cryptosystem implemented and deemed practical for use. It is insufficient to define operations in polynomial time and non-polynomial time when discussing practical implementations of the cryptosystem.

Even though the cryptosystem may be provably secure, if the implementation speeds involved are too slow for an acceptable security parameter, then it is clear that alternatives must be sought. After careful analysis of both lattice-based cryptosystems analyzed in this paper, we found that both have significant space or time complexity drawbacks, possibly preventing widespread adoption.

In this paper we propose a method for improving the speed of Babai's Round-Off CVP approximation algorithm [1] in lattices using the Chinese Remainder Theorem (CRT). We then formulate a new lattice-based cryptosystem with implementation as a prime consideration. This new cryptosystem is based on the work done by Goldreich et al. and Micciancio but has much faster encryption and decryption speeds when implemented on common hardware platforms.

## 2 Lattice Theory and Lattice-Based Cryptography

**Definition 1 (Lattice).** *A lattice $\mathcal{L}$ is a discrete sub-group of $\mathbb{R}^n$, or equivalently the set of all the integral combinations of $d \leq n$ linearly independent vectors over $\mathbb{R}$.*

$$\mathcal{L} = \mathbb{Z}\, b_1 + \cdots + \mathbb{Z}\, b_d, \quad b_i \in \mathbb{R}^n.$$

*$B = (b_1, ..., b_d)$ is called a basis of $\mathcal{L}$ and $d$, the dimension of $\mathcal{L}$.*
*We will refer $\mathcal{L}_B$ as a lattice of basis $B$.*

For a given lattice $\mathcal{L}$, there exists an infinite number of bases. However, the Hermite Normal Form basis (Definition 2) is unique [2].

**Definition 2 (HNF).** *Let $\mathcal{L}$ be an integer lattice of dimension $d$ and $H \in \mathbb{Z}^{d,n}$ a basis of $\mathcal{L}$. $H$ is a Hermite Normal Form basis of $\mathcal{L}$ if and only if*

$$\forall 1 \leq i, j \leq d \quad H_{i,j} \begin{cases} = 0 & \text{if } i > j \\ \geq 0 & \text{if } i \leq j \\ < H_{j,j} & \text{if } i < j \end{cases}$$

The HNF basis can be computed from a given basis in a polynomial time [6].

Many lattice theory problems are based on distance minimization, as determined via the euclidean norm.

**Definition 3 (Euclidean norm).** *Let $w$ be a vector of $\mathbb{R}^n$. The euclidean norm is the function $\|.\|$ defined by $\|w\| = \sqrt{\sum_{i=1}^{n} w_i^2}$.*

There are many different problems based on the minimization of distance in lattice theory. In this paper we focus on the one used by the cryptosystem we analyzed, namely the *Closest Vector Problem.*

**Definition 4 ($\gamma$-CVP).** *Let $w$ be a vector in a lattice $\mathcal{L}$. The Closest Vector Problem is to find a vector $u \in \mathcal{L}, \forall v \in \mathcal{L}, v \neq u, \|w - u\| \leq \gamma \|v - u\|$.*

CVP (for $\gamma = 1$) has been demonstrated to be NP-hard by Emde Boas [3]. However, by limiting ourselves to a special instance of CVP, we will be able to use a good basis to solve CVP using two existing algorithms proposed by Babai [1]. In 1986, Babai proposed two polynomial methods to solve CVP: the *nearest plane* and the *round-off* methods. These algorithms solve CVP within $\gamma = 2^{d/2}$ and $\gamma = 1 + 2d\left(\frac{9}{2}\right)^{d/2}$, respectively. Babai's algorithms use an LLL-reduced basis [7].

## 2.1 Lattice-Based Cryptography

In 1996, Goldreich, Goldwasser and Halevi (GGH) [5] proposed an efficient way to use lattice theory to build a cryptosystem inspired by McEliece cryptosystem [8] and based on the Closest Vector Problem (CVP). Their practical proposition of a cryptosystem was attacked and broken severely by Nguyen in 1999 [10]. However, the general idea is still viable. Until then, the other propositions were made using the same principle [9]. In the following, we briefly review the GGH cryptosystem. A GGH cryptosystem comprises the following algorithms.

- **KeyGenerate:** Compute a "good basis" $A$ and a "bad basis" $B$ of a lattice $\mathcal{L}$, $\mathcal{L}(A) = \mathcal{L}(B)$. Provide $B$ as public and keep $A$ secret.
- **Encrypt:** To encrypt a plaintext vector $p \in \mathbb{Z}^n$: use the bad basis to create a vector $v$ in the lattice $\mathcal{L}$, $v = pB$. Publish the encrypted message which is the addition of this vector with a random error vector $e \in \mathbb{Z}^n$: $c = v + e$.
- **Decrypt:** Use the good basis to find the closest vector in the lattice of the encrypted message $c$. The closest vector of the encrypted message $c$ is the message vector $v$[1]. Using this, obtain the plaintext vector $p$.

The important points for the security and efficiency of those cryptosystems are defined as follows.

i) It is easy to compute a "bad basis" from a "good basis", but it is difficult to compute a "good basis" from a "bad basis".
ii) It is easy to create a random vector of a lattice even with a "bad basis".
iii) It is easy to find the closest vector with a "good basis" but difficult to do so with a "bad basis".

In 2001, Micciancio [9] proposed some major improvements of the speed and the security of GGH. In this scheme, the public key uses a Hermite Normal Form (HNF) for the bad basis. In this scheme, to encrypt a message, we perform a modulo lattice reduction on the plaintext vector (Algorithm 1). In effect, this is done by using the small plaintext vector as the error vector in the GGH scheme, i.e., $c = rB + p$ where $r$ is some minimal vector. The advantage of this scheme is that we are able to use any basis inverse to recover the plaintext and hence do not need to store a transformation matrix, unlike GGH's scheme.

To perform decryption, Micciancio's scheme uses Babai's nearest-plane CVP approximation, as this provides a better approximation than the Round-Off algorithm [1].

## 2.2 Drawbacks of Existing Schemes

**GGH.** A thorough test implementation of the GGH cryptosystem revealed two major shortcomings. Firstly, GGH suffers from slow decryption speeds. Using a theorized computationally secure dimension of 1000, decryption speeds on a conventional, modern PC were around 2 seconds, compared to conventional

[1] Under the supposition that the norm of $e$ is sufficiently small.

**Algorithm 1.** Vector Reduction modulo a HNF Basis

```
Input  : $B \in \mathbb{Z}^{n,n}$ a HNF basis and $p \in \mathbb{Z}^n$ a vector.
Output: $c \in \mathbb{Z}^n$ such that $(c - p) \in \mathcal{L}(B)$ and $\forall i, 0 \leq c_i < B_{i,i}$
begin
    $c \leftarrow p$
    for $i \leftarrow n - 1$ to $0$ do
        $q \leftarrow \lfloor c_i / B_{i,i} \rfloor$
        $c \leftarrow c - q \times B_i$
    end
end
```

cryptosystems operating orders of magnitude faster. This significantly limits the practical applicability of this cryptosystem. Secondly, the storage requirements of public keys for GGH are enormous. In a presumed computationally secure dimension of 1000, public key sizes were seen to be over 290Mb. This is impractical in a public key infrastructure with a large number of clients, such as the Internet or large internal networks.

**Micciancio.** We found that implementing the Micciancio cryptosystem also revealed some shortcomings. Firstly, the use of Nearest-Plane CVP recovery in the Micciancio cryptosystem's decryption phase gave a much lower decryption speed than GGH's round-off CVP recovery method (Figure 1). Secondly, we found that Micciancio requires a large amount of memory to perform key generation, which became larger than the 4Gb maximum on our 32-bit test PC at higher dimensions, preventing key generation at these dimensions on common, consumer hardware.

### 2.3 Chinese Remainder Theorem

The Chinese Remainder Theorem allows the representation of a large, variable-precision integer by its residue modulo some small moduli. This technique is often used to replace arithmetic on large integers with operations over these small moduli.

**Theorem 1 (Chinese Remainder Theorem).** *Let $p_i \in \mathbb{N}$ $n$ coprimes integers, $P = \prod_{i=1}^{n} p_i$ and $P_i = P/p_i$. Then, for any $n$-tuple $a_i$ there exists an unique integer $0 \leq A < P$ such that $a_i = A \bmod p_i$,*

$$A = \sum_{i=1}^{n} a_i (P_i^{-1} \bmod p_i) P_i \bmod P.$$

**Motivation.** We apply the Chinese Remainder Theorem to lattice-based cryptosystems by operating the cryptosystem in an integer ring with an order greater than the largest element, for several reasons. Firstly, performing our calculations in these 'small' finite fields allows us to take advantage of the various platform optimizations for integer arithmetic, since working with variable precision integers larger than the implementation machine's word-size imposes a significant overhead. Secondly, we reduce the need to rely on using a highly optimized implementation of variable-precision integer and matrix arithmetic, specific for each particular platform. While such libraries are readily available for common PC computing platforms, we felt that providing a platform-independent variable precision optimization would be valuable, especially for embedded platforms such as smart cards and cellular telephones. Thirdly, since each finite field calculation is independent of the others it is indeed possible to perform each finite field operation in parallel. While this has not been implemented in this library due to the lack of thread support in Shoup's NTL library [11], we see no intrinsic property of the cryptosystems discussed that would prevent this from operating as intended.

**Construction.** We can construct a sufficiently sized ring in the following way. First, we assess the bound of the size of the coefficients to be calculated. If the coefficients involved include negative coefficients, we must double this bound in order to cover these values. Next, we construct many small finite fields of prime order $p < 2^b$, where $b$ is our target platform's word size, until the product of these primes is above our coefficient bound. We perform our calculations independently in these fields and once these have completed, we perform a simple CRT reconstruction to calculate the final value.

A significant hurdle to adopting this approach for all lattice problems, however, is that it is often difficult to find an appropriate tight bound of the size of the coefficients involved. Due to this, we design our new scheme around this problem, creating a private basis with which we are able to calculate a satisfactorily tight bound on the coefficients to allow for faster key generation.

**Memory usage.** Since each finite field is independent, it is possible to decrypt a plaintext via CRT serially with respect to the key, i.e., only loading each matrix over some finite field into memory as we require it. This has great benefits for memory utilization especially for embedded systems as we are only required to store a matrix of standard integers in primary memory at any given time rather than a matrix of much larger variable-precision integers. In the case of lattice dimension 1000, it can be seen that the memory usage would only be approximately 4Mb at any given time. Obviously in such a case, decryption speeds would be I/O bound in the case of loading each matrix in from a hard drive or flash-based storage.

### 2.4 Improvement of GGH Using CRT

We were able to optimize GGH via CRT by placing $R^{-1}$ into small finite fields and performing the multiplication inside Babai's Round-Off step over these fields. This not only yielded significantly faster decryption speeds, but also provided the platform flexibility discussed above.

Firstly, we define the following scalar function which will be used in our CRT reconstruction.

$$Q(p_i) = \frac{P}{p_i}, P = \prod p_i$$

Secondly, to avoid rational arithmetic, we multiply $R^{-1} \in \mathbb{Q}^{n,n}$ by $\det R$ to obtain $S \in \mathbb{Z}^{n,n}$ ($R^{-1} = S/\det R$). We define a function $R'$ to represent the following:

$$R'(p_i) = R^{-1} \times \det R \times Q(p_i)^{-1} \bmod p_i$$

Since we will be multiplying by $Q(p)^{-1}$ in the reconstruction phase, we are able to save crucial decryption time by precomputing this value in the Key Generation phase.

**Algorithm 2.** CVP Round-Off using CRT

```
Input  : $v \in \mathbb{Z}^n$ the input vector, $R \in \mathbb{Z}^{n,n}$ a basis of $\mathcal{L} \subseteq \mathbb{Z}^{n,n}$.
Output: $w \in \mathcal{L}$ a close vector of $v$ in the lattice $\mathcal{L}$
begin
    $x \leftarrow 0$
    foreach $p_i$ do  $x \leftarrow x + Q(p_i) \times (c \times R'(p_i) \bmod p_i) \bmod P$
    $w \leftarrow \lfloor x/det(R) \rceil \times R$
end
```

Due to the complexity of Micciancio's Nearest-Plane CVP decryption method, we were unable to modify this to work in a CRT environment.

## 3 New Scheme

We design this new scheme specifically to provide faster operations, in a particular faster implementation, while still maintaining a similar structure to existing cryptosystems. Specifically, this involved consideration in the design for decryption using the CRT round-off method discussed earlier.

### 3.1 Key Generation

**Private Basis.** To create the private basis, we use GGH's private basis construction, namely $R \leftarrow bI + M$. This was chosen over Micciancio's basis construction for two reasons. Firstly, it allows us to make generalizations about the bound on the size of $||R||_\infty$ which allows for much faster key generation as we do not need to perform a full matrix inversion. Secondly, and perhaps more importantly, it provides a more orthogonal basis with which to perform decryption, which in turn, decreases the size necessary to ensure correct decryption using CVP Round-off. This in turn allows us to decrease the size of the coefficients while keeping the same security parameter, saving storage and transmission space and increasing efficiency.

**Public Basis.** To create the public basis, we use Micciancio's method of applying a HNF reduction on the private basis, as this provided a greater level of security, simplified key storage and much smaller public keys. Optionally, an LLL reduction can also be applied to this key to reduce the size of the coefficients, however we found this step to be unnecessary.

**Precomputation.** We are able to significantly speed up the decryption phase by precomputing all the required values of the functions $Q(p)$ and $R'(p)$ in the key generation phase and storing these values in a look-up table. This increases the speed of the decryption step, at the expense of a lower key generation speed.

**Algorithm 3.** KeyGenerate

```
Input  : $n \in \mathbb{N}$ the security parameter.
Output: $B \in \mathbb{Z}^{n,n}$ the public key, $R \in \mathbb{Z}^{n,n}$ the private key.
begin
    $M \leftarrow 0$ for $i, j \leftarrow 0$ to $n-1$ do $M_{i,j} \leftarrow Rand(-1, 1)$
    $b \leftarrow \left\lceil 2\sqrt{2n/3} \right\rceil$ repeat $b \leftarrow b + 1$ until $||(bI + M)^{-1}||_\infty \leq 1/2$
    $R \leftarrow bI + M$
    $B \leftarrow HNF(R)$
end
```

## 3.2 Encryption

For encryption, we use Micciancio's method of modulo lattice reduction with the public basis. We felt that this provided excellent speed and provided strong notions of security. We modified Micciancio's construction by limiting the encryption vector domain to {-1, 0, 1} and by using $\|R^{-1}\|_\infty < 1/2$, we can ensure (See [5]) that there will not be any decryption error.

## 3.3 Decryption

Decryption is of a similar form to the improved GGH decryption method using CRT except with a minor change to reflect the encoded message being in the error vector rather than the lattice point. i.e. given the lattice vector $w$, we calculate the plaintext $p = c - w$.

## 4 Implementation and Performance Analysis

Since we are primarily concerned with the implementation aspects of the cryptosystems discussed, we have coded both existing cryptosystems as well as our new scheme in C++ using Victor Shoup's NTL [11] compiled against GNU MultiPrecision Library (GMP) [4]. We feel that these libraries are the most appropriate choice for implementation as they are created with runtime speed a major factor in the design while maintaining numerical stability and correctness. With this in mind, the authors feel that this choice of implementation forms a good basis for comparison. These values were obtained on a 2.1Ghz Intel Core 2 Duo platform with 4Gb RAM.

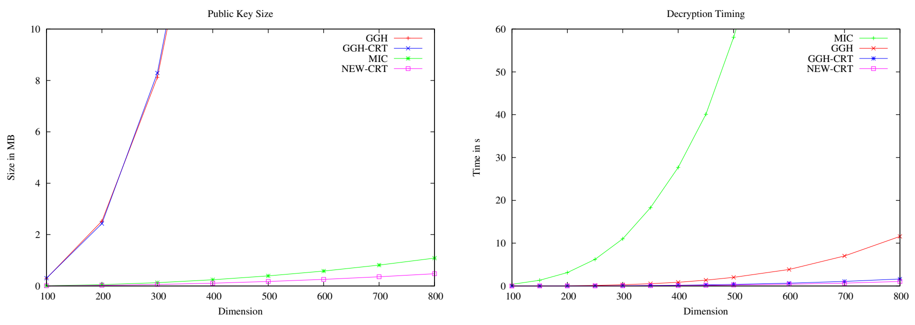

**Fig. 1.** Performance Results

**Table 1.** Speeds and keysizes

| Dimension | 400 | | | | 800 | | | |
|---|---|---|---|---|---|---|---|---|
| | Enc. | Dec. | Pub. Key | Priv. Key | Enc. | Dec. | Pub. Key | Priv. Key |
| GGH | 0.04s | 0.86s | 18.7 MB | 43.0 MB | 0.23s | 11.57s | 153.0 MB | 374.4 MB |
| Micciancio | 0.01s | 27.69s | 241.4 kB | 95.4 MB | 0.02s | 259.94s | 1.1 MB | 861.6 MB |
| GGH (CRT) | 0.04s | 0.18s | 19.3 MB | 72.3 MB | 0.23s | 1.63s | 150.9 MB | 606.4 MB |
| New Scheme | 0.01s | 0.11s | 109.8 kB | 97.0 MB | 0.02s | 1.05s | 476.3 kB | 848.0 MB |

## References

1. Babai, L.: On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica 6, 1–13 (1986)
2. Cohen, H.: A course in computational algebraic number theory. Graduate Texts in Mathematics, vol. 138. Springer, Heidelberg (1993)
3. Van Emde Boas, P.: Another NP-complete Problem and the Complexity of Computing Short Vectors in Lattices. Tech. rep. 81-04, University of Amsterdam (1981)
4. GNU Multiple Precision Arithmetic Library, `http://gmplib.org`
5. Goldreich, O., Goldwasser, S., Halevi, S.: Public-Key Cryptosystems from Lattice Reduction Problems. Tech. rep., Massachusetts Institute of Technology (1996)
6. Kannan, R., Bachem, A.: Polynomial Algorithms for Computing the Smith and Hermite Normal Forms of an Integer Matrix. J. of Computing 8, 499–507 (1979)
7. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 513–534 (1982)
8. McEliece, R.J.: A Public-Key Cryptosystem Based On Algebraic Coding Theory. Deep Space Network Progress Report 44, 114–116 (1978)
9. Micciancio, D.: Improving Lattice Based Cryptosystems Using the Hermite Normal Form. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 126–145. Springer, Heidelberg (2001)
10. Nguyên, P.Q.: Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto 1997. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 288–304. Springer, Heidelberg (1999)
11. Shoup, V.: NTL: A Library for doing Number Theory, `http://www.shoup.net/ntl`