Cryptosystems Based on Lattices

We present some cryptosystems whose common feature is that they all rely on computational problems in lattices for their security. The subject of lattice based cryptography is very active and there have recently been new ideas that revolutionised the field. It is beyond the scope of this book to survey these recent developments.

19.9 The Goldreich-Goldwasser-Halevi Cryptosystem and Variants

The Goldreich-Goldwasser-Halevi (GGH) cryptosystem relies on the difficulty of the closest vector problem (CVP) in a lattice. The system is reminiscent of the McEliece cryptosystem, which we briefly recall in the next paragraph. Encryption for both systems is randomised.

In the McEliece cryptosystem one chooses an error correcting code (some references for error correcting codes are van Lint [391] and Chapter 18 of [609]) over a finite field $\mathbb{F}_q$ (typically $\mathbb{F}_2$) given by a $k \times n$ generator matrix G (where $k < n$) and publishes a "disguised" version $G' = SGP$ where S and P are suitable invertible matrices (we refer to Section 8.5 of Menezes, van Oorschot and Vanstone [418] for details). The public key is G' and the private key is (S, G, P). To encrypt a message $\underline{m} \in \mathbb{F}_q^k$ one computes $\underline{c} = \underline{m}G' + \underline{e}$ where $\underline{e} \in \mathbb{F}_q^n$ is a randomly chosen error vector of low Hamming weight; note that this computation is over $\mathbb{F}_q$. To decrypt one uses the decoding algorithm for the error correcting code.

The basic GGH public key encryption scheme is similar; we give an informal sketch of the idea now. One chooses a "nice" basis B for a full rank lattice $L \subset \mathbb{Z}^n$ and publishes a "disguised" basis $B' = UB$ for L where U is "random" unimodular matrix. A message $\underline{m} \in \mathbb{Z}^n$ is encrypted as $\underline{c} = \underline{m}B' + \underline{e}$ where $\underline{e}$ is a randomly chosen short error vector; note that this computation is over $\mathbb{Z}$. To decrypt one solves the closest vector problem, using the nice basis B, to obtain the lattice point $\underline{m}B'$ close to $\underline{c}$; one can then obtain $\underline{m}$.

While encryption is superficially the same for the McEliece and GGH cryptosystems, there are significant differences between the security analysis of these schemes. An advantage of the lattice approach is that the error vector is required to have less structure: it is only required to be short, compared with McEliece where the error vector must have low Hamming weight. Both schemes have ciphertexts larger than the messages but an advantage of McEliece is that ciphertexts have a fixed size whereas for GGH the coefficients are integers whose size can vary significantly.

Exercise 19.9.1. Show that any cryptosystem based on the McEliece or GGH idea does not have indistinguishability security under passive attack.

Exercise 19.9.2. A variant of the McEliece or GGH proposal is to swap the roles of the message and the randomness. In other words one encodes the message as a valid error vector $\underline{m}$, chooses a random $\underline{e} \in \mathbb{F}_q^k$ (respectively, $\underline{e} \in \mathbb{Z}^k$) and computes $\underline{c} = \underline{e}G' + \underline{m}$ (resp., $c = \underline{e}B' + \underline{m}$). Show that this variant also does not have indistinguishability security under passive attacks.

Exercise 19.9.3. Show that any cryptosystem based on the McEliece or GGH idea (and without any padding scheme) does not have one way encryption security under a CCA attack.

Exercises 19.9.1 and 19.9.3 show that the 'textbook' (i.e., without a padding scheme) McEliece and GGH cryptosystems should not be used in practice. Using techniques similar to those presented later for Elgamal and RSA one can prevent such attacks as long as the basic scheme is OWE-CPA secure (see Section 1.3.1 for the definition of this security notion). Hence, for the rest of this chapter we focus purely on the textbook versions and mainly consider security under passive attacks.

Exercise 19.9.4. Given a GGH public key B' show that there is more than one private basis matrix B that is suitable for decryption. Show that one can efficiently determine whether a guess B for a GGH private basis matrix can correspond to a given public basis B'.

To give a precise definition of the GGH cryptosystem it is necessary to give the following details:

1. What dimension n should be used?
2. How does one choose the "nice" lattice basis B and what properties should it have?
3. How does one choose the "random" unimodular matrix U?
4. What is the message space and how does one encode information into a vector $\underline{m}$?
5. How does one choose the error vector $\underline{e}$ and in what space does it lie?

We briefly sketch the proposal of Goldreich, Goldwasser and Halevi; the reader should refer to the original paper [256] for complete details. It is suggested to take $n \geq 200$. Two methods to generate the "nice" basis B are given: one is to choose a random matrix B with entries in, say, $\{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$ (so that all vectors are relatively short); another is to choose $B = kI_n + E$ where I_n is the $n \times n$ identity matrix, $k > 1$ is a "medium sized" integer and E is a random matrix with small entries as in the first case above. Two methods to generate U are also given; in both cases the issue is to ensure that the coefficients of $B' = UB$ do not explode in size (we refer to Section 3.2 of [256] for details). The message space is the set of vectors of length n with entries in

$\{-M, -(M-1), \ldots, -1, 0, 1, \ldots, M-1, M\}$ for some $M \in \mathbb{N}$ ([256] actually suggests $M = n$). Finally, the error vector is chosen to be a random vector of length n with entries in $\{-\sigma, \sigma\}$ for some $\sigma \in \mathbb{N}$ (typically, $\sigma = 3$).

As mentioned above, to encrypt a message $\underline{m}$ one computes the ciphertext $\underline{c} = \underline{m}B' + \underline{e}$. To decrypt $\underline{c}$ one uses the Babai rounding technique with respect to the nice basis B for the lattice. More precisely, multiply $\underline{c}$ by B^{-1} to obtain

$$\underline{c}B^{-1} = (\underline{m}B' + \underline{e})B^{-1} = \underline{m}UBB^{-1} + \underline{e}B^{-1} = \underline{m}U + \underline{e}B^{-1} \in \mathbb{Q}^n.$$

The Babai rounding will remove the term $\underline{e}B^{-1}$ as long as it is small enough. One then multiplies by U^{-1} to get the message $\underline{m}$.

Exercise 19.9.5. Write down algorithms for KeyGen, Encrypt and Decrypt.

Example 19.9.6. Let $L \subset \mathbb{R}^2$ be the lattice with basis matrix

$$B = \begin{pmatrix} 17 & 0 \\ 0 & 19 \end{pmatrix}.$$

Let

$$U = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix} \quad \text{giving} \quad B' = UB = \begin{pmatrix} 34 & 57 \\ 51 & 95 \end{pmatrix}.$$

Let the message be $\underline{m} = (2, -5)$ and take $\underline{e} = (1, -1)$ (this is GGH encryption with $\sigma = 1$). Then

$$\underline{c} = \underline{m}B' + \underline{e} = (-186, -362).$$

To decrypt one computes

$$\underline{c}B^{-1} \approx (-10.94, -19.05)$$

(note that $\underline{m}U = (-11, -19)$ and $\underline{e}B^{-1} \approx (0.06, -0.05)$). We round the above to $(-11, -19)$ and recover the message as $\underline{m} = (-11, -19)U^{-1} = (2, -5)$.

Exercise 19.9.7. Show that GGH decryption gives the correct result as long as the entries of $\underline{e}B^{-1}$ are real numbers of absolute value $< 1/2$. Let ρ be the maximum, in the ℓ_1-norm, of the columns of B^{-1}. Show that if $\sigma < 1/(2\rho)$ then decryption gives the correct result.

Exercise 19.9.8. For the public key in Example 19.9.6 decrypt the ciphertext $\underline{c} = (220, 400)$.

As mentioned, the ciphertext in GGH encryption is considerably larger than the message. A precise analysis of this depends on the sizes of entries in B' (which in turns depends on the specific choices for B and U). We do not give any estimates for the ciphertext expansion.

Micciancio [421] proposed a variant of the GGH cryptosystem. The first idea is, instead of choosing the public basis to be $B' = UB$ for a random matrix $U \in \mathrm{SL}_2(\mathbb{Z})$, one can choose B' to be the Hermite normal form (HNF) of B. There is no loss of security by doing this, since anyone can compute the HNF of UB, and get the same result. The second idea is to encode the message in the error vector rather than in the lattice point (this is the same idea as discussed in Exercise 19.9.2) and to reduce it to the orthogonalized parallelepiped (see Exercise 19.9.9). This results in significantly shorter ciphertexts than the original GGH system and makes the encryption process deterministic. We refer to [421] for further details.

Exercise 19.9.9. Let $\underline{b}_1, \ldots, \underline{b}_n$ be an ordered basis for a lattice L and let $\underline{b}_1^*, \ldots, \underline{b}_n^*$ be the corresponding Gram-Schmidt vectors. Define the **orthogonalized parallelepiped**

$$\mathcal{P} = \left\{ \sum_{i=1}^{n} x_i \underline{b}_i^* : 0 \leq x_i \leq 1 \right\}.$$

Given $\underline{v} \in \mathbb{R}^n$ show how to compute $\underline{w} \in \mathcal{P}$ such that $\underline{v} - \underline{w} \in L$. This is called reducing to the orthogonalized parallelepiped.

19.10 Cryptanalysis of GGH Encryption

We now discuss the one-way encryption (OWE) security of the GGH cryptosystem under passive attacks. There are three natural ways to attack the GGH cryptosystem:

1. Try to obtain the private key B from the public key B'.
2. Try to obtain information about the message from the ciphertext, given that the error vector is small.
3. Try to solve the CVP of $\underline{c}$ with respect to the lattice L defined by B'.

We also present a fourth attack, due to Nguyen, which exploits the particular format of the error vectors in the GGH cryptosystem. Lattice basis reduction algorithms have a role to play in the first and third of these attacks.

Computing a Private Key

For the first attack, we simply run a lattice basis reduction algorithm (such as LLL) on the public basis matrix B'. If we are lucky then it will output a basis B'' that is good enough to allow the efficient solution of the required closest vector instances.

Example 19.10.1. Let

$$B = \begin{pmatrix} 7 & 0 & 0 \\ 0 & 23 & 0 \\ 0 & 0 & 99 \end{pmatrix}$$

and define $B' = UB$ where

$$U = \begin{pmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ -11 & 5 & 1 \end{pmatrix} . \begin{pmatrix} 1 & 3 & -10 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then B' is the public basis matrix

$$B' = \begin{pmatrix} 7 & 69 & -990 \\ 56 & 575 & -8514 \\ -77 & -644 & 8019 \end{pmatrix}.$$

Let $\underline{m} = (2, -1, 3)$ be a message and $\underline{e} = (-1, 1, 1)$ an error vector and define $\underline{c} = \underline{m}B' + \underline{e} = (-274, -2368, 30592)$. Running LLL on B' does yield (up to sign) the matrix B (and hence $U = B'B^{-1}$). From this one can recover $\underline{m}$.

To prevent such an attack it is necessary that the dimension of the lattice be sufficiently large.

Computing Information about the Message

For the second attack we exploit the fact that $\underline{c} = \underline{m}B' + \underline{e}$ where $\underline{e}$ is a vector with small entries. A naive attack is to try all values of the error vector $\underline{e}$ until $\underline{c} - \underline{e}$ lies in the image of $\mathbb{Z}^n B'$. A more subtle idea is to compute $\underline{c}(B')^{-1} = \underline{m} + \underline{e}(B')^{-1}$ and try to deduce possible values for some entries of $\underline{e}(B')^{-1}$. For example, if the j-th column of $(B')^{-1}$ has particularly small norm then one can deduce that the j-th entry of $\underline{e}(B')^{-1}$ is always small and hence get an accurate estimate for the j-th entry of $\underline{m}$. We refer to Section 4.2 of [256] for further discussion. To defeat this attack one should not naively encode the message as a vector $\underline{m} \in \mathbb{Z}^n$. Instead, one should only use some low-order bits of some entries of $\underline{m}$ to carry information, or use an appropriate randomised padding scheme.

Solving the CVP Directly

For the third attack one can consider any of the algorithms listed in Chapter 18 for solving the CVP. For example, one can use the Babai nearest plane algorithm or the embedding technique.

Example 19.10.2. We use the public key and ciphertext from Example 19.10.1 and recover the message using the embedding technique. Construct

$$A = \begin{pmatrix} 7 & 69 & -990 & 0 \\ 56 & 575 & -8514 & 0 \\ -77 & -644 & 8019 & 0 \\ -274 & -2368 & 30592 & 1 \end{pmatrix}.$$

Running LLL on A yields the matrix

$$\begin{pmatrix} -1 & 1 & 1 & 1 \\ 5 & 2 & 2 & 2 \\ -1 & -15 & 8 & 8 \\ -1 & -2 & 51 & -48 \end{pmatrix}.$$

As desired, the first row is $(-1, 1, 1, 1) = (\underline{e}, 1)$. From this one can compute the message as $\underline{m} = (\underline{c} - \underline{e})(B')^{-1}$.

Exercise 19.10.3. For the public key from Example 19.10.1 use the embedding technique to decrypt the ciphertexts $\underline{c} = (120, 1220, -18017)$ and $\underline{c} = (-83, -714, 9010)$.

To defeat such attacks it is necessary that the lattice dimension is sufficiently large and that the solution to the CVP instance is not too special. In particular, the error vector should not be too short compared with the vectors the lattice.

Nguyen's Attack

Nguyen noted that the choice of the error vector in the original GGH cryptosystem made it extremely vulnerable to attack. Write $\underline{\sigma} = (\sigma, \sigma, \ldots, \sigma) \in \mathbb{Z}^n$. The crucial observation is that if $\underline{c}$ is a GGH ciphertext then $\underline{c} + \underline{\sigma} \equiv \underline{m}B' \pmod{2\sigma}$. If B' is invertible modulo 2σ (or even modulo a factor of 2σ) then one can already extract significant information about the message $\underline{m}$. Furthermore, if one successfully computes $\underline{m}_0 \equiv \underline{m} \pmod{2\sigma}$, then one obtains the simpler closest vector instance

$$\frac{\underline{c} - \underline{m}_0 B'}{2\sigma} = \underline{m}' B' + \frac{\underline{e}}{2\sigma}$$

where $\underline{m} = \underline{m}_0 + 2\sigma\underline{m}'$. Since $\underline{e}/(2\sigma)$ is a much shorter vector than $\underline{e}$ it is possible that algorithms for the closest vector problem that were not successful on the original instance can succeed on the new instance.

Example 19.10.4. Consider the lattice L and ciphertext c from Example 19.9.6. Since $\sigma = 1$ we can add $(1,1)$ to c and solve

$$c + (1,1) = (-185, -361) \equiv \underline{m}_0 B' \pmod 2$$

(note that B' is invertible over $\mathbb{F}_2$). One finds $\underline{m}_0 = (0,1) \equiv (2,-5) \pmod 2$ as expected.

Exercise 19.10.5. Perform Nguyen's attack for the ciphertexts of Exercise 19.10.3.

The natural approach to resist Nguyen's attack is to choose error vectors with a more general range of entries (e.g., $e_j \in \{-\sigma, -(\sigma-1), \ldots, -1, 0, 1, \ldots, \sigma\}$ for $1 \leq j \leq n$). It is then necessary to re-evaluate all the other attacks and parameter choices.

Finally, we remark that none of the above techniques gives an attack with polynomial asymptotic complexity as the dimension n grows. Hence, the GGH encryption scheme and its variants are not broken. On the other hand, in practice one needs to use lattices of rather large dimension and this limits the practicality of the GGH system.

19.11 GGH Signatures

Let B' be a GGH public key corresponding to a lattice L in $\mathbb{Z}^n$. The natural signature scheme is as follows: Given a message m hash it to a "random" element $H(m) \in \mathbb{Z}^n$. Then, using the private key, compute a lattice vector $\underline{s}$ close to $H(m)$. The signature on message m is then $\underline{s}$. To verify the signature one checks that $\underline{s}$ lies in the lattice (i.e., $\underline{s}(B')^{-1} \in \mathbb{Z}^n$) and that $\|\underline{s} - H(m)\|$ is smaller than some threshold that is specified as part of the signature verification key.

We remark that signatures for lattice schemes are somewhat easier than for the McEliece cryptosystem since CVP algorithms work for any point in $\mathbb{R}^n$ whereas decoding algorithms may fail for words that are not within the minimum distance of a code-word (however, see Courtois, Finiasz and Sendrier [451] for a study of McEliece signatures).

To analyse the security (namely, resistance to forgery) of such a signature scheme one must consider all the attacks mentioned above on the encryption scheme. One therefore is required to use lattices of large dimension $n \geq 200$.

Furthermore, as usual with signatures, one must also consider the fact that an adversary could obtain signatures on messages and that this might leak information about the private key. For the GGH signature scheme one sees that $\underline{s} - H(m)$ is a short vector in $\mathbb{R}^n$. Indeed, if the CVP algorithm used by the signer is perfect for the basis B then $\underline{s} - H(m)$ always lies in the parallelepiped

$$\mathcal{P}_{1/2}(B) = \{\underline{x}B : \underline{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n, -1/2 \leq x_i \leq 1/2 \text{ for all } 1 \leq i \leq n\},$$

which is called a **fundamental domain** for the lattice (i.e., for every point $\underline{x} \in \mathbb{R}^n$ there is some $\underline{y}$ in the lattice such that $\underline{x} - \underline{y} \in \mathcal{P}_{1/2}(B)$). The fundamental domain of a lattice is a simplex whose sides are determined by the basis vectors in B. Hence, it is natural to wonder whether seeing a number of random entries in $\mathcal{P}_{1/2}(B)$ allows one to learn something about the vectors in B. Nguyen and Regev [457, 458] have explored this idea and shown that such an approach can be used to cryptanalyse signatures. Adding a "perturbation" to the signature seems to prevent the attack of Nguyen and Regev (see Section 1.3 of [458]). Gentry, Peikert and Vaikuntanathan [253] give a method to sample