COS 533: Advanced Cryptography — Princeton University
Lecture 17 (November 15, 2017)
Lecturer: Mark Zhandry — Scribe: Enric Boix

Notes for Lecture 17

1 Last Time

Last time, we defined the Short Integer Solution (SIS) and Learning With Errors (LWE) problems. These are related to the SVP and CVP problems on lattices, and are conjectured to be hard for certain parameter ranges.

In the following notes, we will show how to build (1) a signature scheme and (2) an identity-based encryption scheme (IBE) assuming the hardness of SIS and of LWE, respectively. Additionally, we will (3) begin to explain how to build a fully-homomorphic encryption scheme (FHE) based on the hardness of LWE.

2 Signature Scheme

We will need the following two useful theorems:

Theorem 1 ([Ajt99]) *Given n, m, q, one can efficiently generate a matrix $A \in \mathbb{Z}_q^{n\times m}$ together with $T \in \mathbb{Z}_q^{m\times m}$ such that*

1. *$A \cdot T \equiv 0 \pmod q$.*
2. *"T is short." In particular, $|T| \leq m^{O(1)}$.*
3. *A is statistically close to random.*
4. *T is invertible over $\mathbb{Z}$.*

Theorem 2 ([GPV08]) *There is an efficient algorithm $\mathsf{sampleD}(T, \sigma, c)$, which, given $T \in \mathbb{Z}_q^{m\times m}$, $c \in \mathbb{R}^m$, and $\sigma > 0$, samples from a distribution statistically close to the discrete Gaussian $D_{L(T),\sigma,c}$ as long as*

$$\sigma \geq |T|\omega(\sqrt{\log m}).$$

Recall the definition of the discrete Gaussian

$$\Pr[x \in D_{L(T),\sigma,c}] \propto e^{-\pi|x-c|^2/\sigma^2},$$

for all $x \in L(T)$, which is the m-dimensional lattice defined by T.

Definition of signature scheme

Now we can define a signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, due to [GPV08]. Assume that everyone has evaluation access to a random-looking hash function on the messages $H : \mathcal{M} \to \mathbb{Z}_q^n$:

1. $\mathsf{Gen}()$ generates $(A \in \mathbb{Z}_q^{n \times m}, T \in \mathbb{Z}_q^{m \times m})$ randomly as in Theorem 1. $(\mathsf{pk}, \mathsf{sk}) = (A, T)$.
2. $\mathsf{Sign}(A, T, x)$ samples a "short" vector $s \in \mathbb{Z}^m$ such that $H(x) \equiv A \cdot s \pmod q$. This is done as follows:
   (a) Use linear algebra to find a solution s_0 to $H(x) \equiv A \cdot s_0 \pmod q$.
   (b) Since T is short and full-rank over $\mathbb{Z}$ and $A \cdot T \equiv 0 \pmod q$, we use $\mathsf{sampleD}(T, \sigma, -s_0)$ to sample
   $$v \xleftarrow{\$} D_{\lambda_q^\perp(A), \sigma, -s_0}.$$
   (c) Output $s = s + v \pmod q$, which is short because v is close to $-s_0$. Moreover,
   $$A \cdot s = (A \cdot s_0) + (A \cdot v).$$
   The first term is $H(x)$ modulo q, and the second vanishes modulo q because $v \in \Lambda_q^\perp(A)$. Therefore
   $$A \cdot s \equiv H(x) \pmod q.$$
3. $\mathsf{Ver}(A, x, s)$ checks that $H(x) \equiv A \cdot s \pmod q$.

EUF-CMA security of signature scheme

We do not know a proof of security that relies on concrete properties of the hash function H (e.g., collision-free, one-way, etc...). Instead, we pretend that H is a truly random function to which anyone can make queries (Random Oracle Model). We show that if SIS hardness holds, then the scheme is secure.

The proof is by a hybrid argument. We construct two experiments that are statistically close (although we will not show their closeness formally).

Experiment 1 is the standard EUF-CMA security experiment for the scheme. That is, the challenger runs $(A, T) \leftarrow \mathsf{Gen}()$ and gives A to the adversary at the very beginning. Then, the adversary is allowed to make two different kinds of queries as many times as it would like:

1. Message-signing queries: the adversary sends x to the challenger and receives a signature $s \leftarrow \mathsf{Sign}(T, x)$.

2. Random oracle queries: the adversary sends x to the challenger and receives the value of $H(x)$.

Finally, the adversary returns a pair (x^*, s^*). It wins if $H(x^*) = A \cdot s \pmod q$, and s^* is short, and the adversary has not made the message-signing query x^* and received the signature s^* for it during the experiment.

Experiment 2 is different only in the behavior of the challenger. Instead of generating (A, T) via Gen, the challenger sends a random $A \in \mathbb{Z}_q^{n \times m}$ to the adversary at the very beginning.

The challenger also generates a truth table $J : \mathcal{M} \to \mathbb{Z}^m$ such that for every message $x \in \mathcal{M}$, $J(x) \sim D_{\mathbb{Z}^m, \sigma, 0}$ i.i.d. This truth table is exponentially large, but the challenger can just generate it on the fly by using the $\mathsf{sampleD}$ algorithm.

Then,

1. for every message-signing query x, the challenger returns $J(x)$.
2. for every random-oracle query x, the challenger returns

$$H(x) := A \cdot J(x) \pmod q.$$

The verification step at the very end is the same as for the first experiment.

Heuristically, the two experiments are statistically close, because (a) by Theorem 1, the public key for Experiment 1 is statistically close to a random matrix, which is the public key for Experiment 2. (b) As long as the dimension m is large enough, H will be close to a uniformly random function in Experiment 2 because for each $x \in \mathcal{M}$, $J(x)$ is independent and has entropy growing in m, and $H(x)$ is the product of a random matrix with $J(x)$. (c) Fixing $H(x)$, the signature s_x returned by Experiment 1 is statistically close to a discrete Gaussian on $\mathbb{Z}^m$ with center 0 and standard deviation σ, conditioned on $H(x) \equiv A \cdot s_x \pmod q$.

Therefore, we just need to show that no adversary can break Experiment 2. Suppose the adversary outputs a valid solution (x^*, s^*) to this experiment. Then, either

1. the adversary has made a message-signing query to x^* before, and received a signature $J(x^*) \neq s^*$ for it. In this case,

$$A \cdot (J(x^*) - s^*) \equiv H(x^*) - H(x^*) \equiv 0 \pmod q,$$

   so $(J(x^*) - s^*) \neq 0$ is a solution to SIS for A.

2. Or the adversary has not made a message-signing query to x^* before. In this case s^* is probably not equal to $J(x^*)$, so $(J(x^*) - s^*) \neq 0$ is probably a SIS solution for A, as in the previous case. The reason for the inequality is a technical

detail whose proof we omit: with high probability over the choice of A, the distribution of $y \xleftarrow{\$} D_{\mathbb{Z}^m,\sigma,0}$ given that $A \cdot y = H(x^*)$ has high min-entropy. And s^* and $J(x^*)$ are independent given $H(x^*)$, since the adversary has not made a message-signing query for x^*.

Therefore, given a random matrix $A \xleftarrow{\$} \mathbb{Z}^{m \times m}$ an adversary can break the average-case hardness of SIS by simulating the challenger and adversary to Experiment 2 with matrix A, and returning $J(x^*) - s^*$. The result will be a SIS solution for A with non-negligible probability.

3 Identity-Based Encryption (IBE)

These lattice-based methods can be applied to create an Identity-Based Encryption (IBE) scheme (also due to [GPV08]). In an IBE, users' IDs (for example their e-mail addresses or their social security numbers) essentially act as their public keys. So in order for Alice to send an encrypted message to Bob, the only thing Alice needs to know about Bob is Bob's ID.

Of course this is not the whole picture. In order to encrypt a message to anyone, Alice also needs to know a master public key that has been generated and disseminated by a central authority (for example, a government). And in order for Bob, say, to decrypt a message sent to him, he needs to know his own private key, which he can get from the central authority if he can prove that he matches his ID.

Definition of IBE scheme

The IBE scheme ($\mathsf{IBEGen}, \mathsf{IBEExtract}, \mathsf{IBEEnc}, \mathsf{IBEDec}$) is based on the signature scheme above, and on the dual of the Regev Encryption Scheme we saw last time. As in the signature scheme, we assume we have access to a random hash function $H : \mathsf{ID} \to \mathbb{Z}_q^m$.

1. $\mathsf{IBEGen}()$ generates $(A \in \mathbb{Z}_q^{n \times m}, T \in \mathbb{Z}_q^{m \times m})$ randomly as in Theorem 1. $(\mathsf{mpk}, \mathsf{msk}) = (A, T)$. This is the same as Gen for the signing procedure.

   The central authority runs IBEGen at the very beginning, and makes mpk public.

2. $\mathsf{IBEExtract}(A, T, id)$ returns $\mathsf{sk}_{id} \leftarrow \mathsf{Sign}(A, T, id)$. sk_{id} is a short vector such that $A \cdot \mathsf{sk}_{id} = H(id)$.

   The central authority runs this to give a user her secret key after it has verified her identity.

3. $\mathsf{IBEEnc}(A, id, b)$ encrypts a bit b as follows

(a) choose $s \xleftarrow{\$} \mathbb{Z}_q^n, x \xleftarrow{\$} D_{\mathbb{Z}^m,\sigma,0}$, and $x_0 \xleftarrow{\$} D_{\mathbb{Z},\sigma,0}$.

(b) Let $p = s^T A + x^T$ and $c = s^T H(id) + x_0 + b\lceil q/2 \rfloor$. This is an LWE instance.

(c) Output (p, c).

4. $\mathsf{IBEDec}(\mathsf{sk}_{id}, p, c)$ decrypts the ciphertext (p, c) as follows: Compute

$$c - p \cdot \mathsf{sk}_{id} = s^T H(id) + x_0 + b\lceil q/2 \rfloor - (s^T A + x^T) \cdot \mathsf{sk}_{id} =$$

$$s^T H(id) + x_0 + b\lceil q/2 \rfloor - s^T H(id) + x^T \mathsf{sk}_{id} = x_0 + x^T \mathsf{sk}_{id} + b\lceil q/2 \rfloor.$$

Since x_0 and $x^T \mathsf{sk}_{id}$ are both short, we can round the answer to get b.

The security proof here is a bit more involved than the last one, and we omitted it in class. The proof takes place under the random oracle model, as before. The general idea is that each ciphertext instance (p, c) is indistinguishable from random by LWE hardness, and that the secret keys sk_{id} are secure by the security of the signature scheme.

4 Fully-Homomorphic Encryption (FHE)

We will now begin our construction of a Fully-Homomorphic Encryption scheme (FHE). The goal of FHE schemes is to allow computers to do arbitrary computation over encrypted data.

For instance, suppose you have a big and sensitive database. You want to use the cloud's computing resources to calculate statistics on this database, and to train your state-of-the-art machine-learning models. You could simply send the database to the cloud, but then you would be giving the cloud access to all of your private data. Or you could encrypt the database using traditional cryptographic methods and send the ciphertext to the cloud, but then the cloud would not be able to do computation on the database. For a long time, there was no known way to get past this privacy-functionality tradeoff.

Enter FHE:

Definition 3 (FHE scheme) *An FHE scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \oplus, \otimes)$ *is an encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *together with addition and multiplication operations* $\oplus, \otimes$ *on the ciphertexts such that for any two ciphertexts* c_0 *and* c_1*,*

$$\mathsf{Dec}(\mathsf{sk}, c_0 \oplus c_1) = \mathsf{Dec}(\mathsf{sk}, c_0) + \mathsf{Dec}(\mathsf{sk}, c_1)$$

and

$$\mathsf{Dec}(\mathsf{sk}, c_0 \otimes c_1) = \mathsf{Dec}(\mathsf{sk}, c_0) \times \mathsf{Dec}(\mathsf{sk}, c_1).$$

Since any computation can be written as computation over a boolean circuit, and hence over an arithmetic circuit, FHE provides a way to do arbitrary computation over encrypted data. Until recently, additively homorphic schemes and multiplicatively homomorphic schemes were known – but no scheme combined both at once.

Notice, for example, that the dual Regev encryption scheme embedded in the IBE scheme above is additively homomorphic (If (p_i, c_i) encrypts i for some secret key sk_{id}, then $(p_0 + p_1, c_0 + c_1)$ encrypts $b_0 + b_1$.) However, the scheme is not multiplicatively homomorphic.

Idea 0: A naive, but illustrative approach to solving FHE is the following:

1. Have our secret key sk be a vector in $\mathbb{Z}_q^m$ and have our ciphertexts be matrices $C \in \mathbb{Z}_q^{n \times m}$.
2. We stipulate that C encrypts x if $Cv = xv$.
3. *Additively homomorphic*: if C_0 encrypts x_0 and C_1 encrypts x_1, then
$$(C_0 + C_1)v = C_0v + C_1v = x_0v + x_1v = (x_0 + x_1)v,$$
so $(C_0 + C_1)$ encrypts $(x_0 + x_1)$.
4. *Multiplicatively homomorphic*: if C_0 encrypts x_0 and C_1 encrypts x_1, then
$$C_0C_1v = C_0(x_1v) = x_1C_0v = x_1x_0v = x_0x_1v,$$
so C_0C_1 encrypts x_0x_1.

However, a scheme with this structure could not be secure, since an adversary can always do an eigendecomposition of ciphertext C to get the values x_i and vectors v_i such that $Cv_i = x_iv_i$.

Idea 1: This leads to a better approach, to add noise to Idea 0 (as in LWE). Then,

1. we can stipulate that C encrypts x if $Cv = xv + e$, where e is a short error vector.
2. *Additively homorphic*: If C_0 encrypts x_0 and C_1 encrypts x_1, then
$$(C_0 + C_1)v = (x_0 + x_1)v + (e_0 + e_1),$$
so $(C_0 + C_1)$ encrypts $(x_0 + x_1)$ as long as the noise $e_0 + e_1$ remains small.
3. *Multiplicatively homorphic*: If C_0 encrypts x_0 and C_1 encrypts x_1, then
$$C_0C_1v = C_0(x_1v + e_1) = x_0x_1v + (x_1e_0 + C_0e_1),$$
so C_0C_1 encrypts x_0x_1 as long as the noise $x_1e_0 + C_0e_1$ remains small.

The problem with this scheme, though, is that in order for the noise terms to remain small, we need the ciphertexts C and the encrypted bits x to remain small as well.

Next time, we will see how to use the $n \times nr$ gadget matrix

$$G = \begin{bmatrix} 1 & 2 & 4 & \dots & 2^r & & & & & & & \\ & & & & & 1 & 2 & 4 & \dots & 2^r & & \\ & & & & & & & & & & 1 & \dots \\ & & & & & \vdots & & & & & & \end{bmatrix},$$

where $r = \lceil \log q \rceil$.

Notice that given $v \in \mathbb{Z}_q^n$, we can compute short $u \in \{0,1\}^{nr}$ such that $Gu \equiv v \pmod{q}$, simply by using the binary representation of v.

This matrix will help keep the ciphertext sizes down when we multiply.

References

[Ajt99] Miklós Ajtai. Generating hard instances of the short basis problem. In *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings*, pages 1–9, 1999. 1

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206, 2008. 1, 2, 4