MAD 4471: Introduction to Cryptography and Coding Theory **Fall 2022**

Lecture 18: KEM based on Module-LWE

Lecturer: Jean-François Biasse *TA: William Youmans*

In this lecture, we present the Module-LWE problem, which is a generalization of the LWE problem where elements are in $\mathbb{Z}_q[X]/X^n+1$ rather than $\mathbb{Z}_q$. We define Key Encapsulation Mechanisms (KEM), and we show that a KEM can be described from the Module-LWE problem. Such a strategy is the basis of the National Institute for Standard and Technology (NIST) standard Kyber. We start be defining KEMs. Then we describe the LWE problem, and we show how it can be attacked by searching for a short vector in a Euclidean lattice. Finally, we show how a CPA-secure encryption scheme can be designed, and we show how to turn this into a KEM.

18.1 Key Encaspulation Mechanism (KEM)

A KEM is a protocol that allows Alice and Bob to have a shared key to be used with a symmetric protocol. Besides the Diffie-Hellman protocol, we do not know of many examples where the key is created in one go. Another way to proceed is to have Alice create a symmetric key, and then transmit it to Bob. The components of a KEM are the following:

- Key generation: outputs a pair Pk, Sk.
- Encapsulation: $\mathrm{Encaps}(\mathrm{Pk}) = c, K$ a ciphertext and a key.
- Decapsulation: $\mathrm{Decaps}(c, \mathrm{Sk}) = K$ a key.

To share a key with Bob, Alice creates c, K using Encaps(.) on input Bob's public key, and sends him the output c. Then Bob recovers K by using Decaps(.) on input c and his private key Sk. We say that the scheme is *correct* if Bob retrieves Alice's key K with overwhelming probability.

Example 1 *Assume that we have a CCA-secure public key encryption scheme given by* $\mathrm{Enc}, \mathrm{Dec}$. *For example, RSA with OAEP. Then we can define the following KEM:*

- *Encapsulation: Draw K uniformly at random in the key space $\mathcal{K}$ and produce $c = \mathrm{Enc}(K, \mathrm{Pk})$.*
- *Decapsulation: Recover $K = \mathrm{Dec}(c, \mathrm{Sk})$.*

A typical notion of security for a KEM is indistinguishability under Chosen Ciphertext Attacks. In the corresponding security game, an adversary is given the public key, a ciphertext, and a key. The goal of the adversary is to determine whether the the ciphertext is the encapsulation of the key, or some random key from $\mathcal{K}$. Additionally, the adversary is allowed to make decapsulation queries for ciphertexts different than the one they received from the challenger.

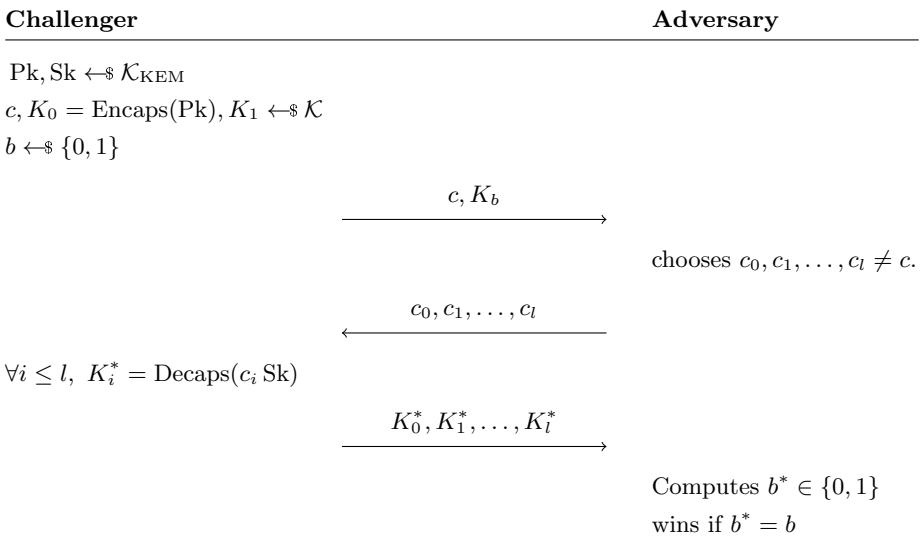

Figure 18.1: Indistinguishability under Chosen Ciphertext Attacks

18.2 Module-LWE

To understand Module-LWE, we first recall the definition of LWE. Let $q > 0$ be a modulus, $m, k > 0$ be integers[1], σ be a standard deviation, and χ_σ be the rounded Gaussian distribution over $\mathbb{Z}_q$ with standard deviation σ. Given m samples of the form

$$\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i,$$

where $\mathbf{a}_i$ is distributed uniformly at random in $\mathbb{Z}_q^k$ and $e_i \leftarrow \chi$, find the secret $\mathbf{s} \in \mathbb{Z}_q^k$.

The Module-LWE problem is defined similarly, except that the $\mathbf{a}_i$ are taken in $\left(\mathbb{Z}_q[X]/X^n + 1\right)^k$ and $\mathbf{e}$ is in $\left(\mathbb{Z}_q/X^n + 1\right)^m$ where n is an integer (typically a power of 2). When $n = 1$, we fall back on the traditional LWE problem. When $k = 1$, we have the so-called *Ring-LWE* problem. With Module-LWE, the difficulty of the problem can be adjusted by moving n or k (and of course q). Since arithmetic can be heavily optimized, practical Module-LWE solutions deployed typically fix q and n to make sure that arithmetic in $\mathbb{Z}_q/X^n + 1$ is optimized once for all, and they increase k to drive the hardness up. Typically, k takes moderate values ($k = 2, 3$ for ex.).

Example 2 *Let $q = 11$,$n = 3$, $k = 2$, and $m = 6$. Let the secret $\mathbf{s}$ be*

$$\mathbf{s} = (X + 1, X - 1),$$

where we identify polynomials $P(X)$ with their residue class in $\mathbb{Z}_q/X^n + 1$. An example of a challenge for the Module LWE problem is given below:

- *$i = 1$: $\mathbf{a}_1 = (X, X^2)$, $e_1 = 1$, $\langle \mathbf{a}_1, \mathbf{s} \rangle + e_1 = 1$.*
- *$i = 2$: $\mathbf{a}_2 = (X^2, X + 1)$, $e_2 = X$, $\langle \mathbf{a}_2, \mathbf{s} \rangle + e_2 = 2X^2 + X - 2$.*
- *$i = 3$: $\mathbf{a}_3 = (5, 3X^2 + 3)$, $e_3 = 2$, $\langle \mathbf{a}_3, \mathbf{s} \rangle + e_3 = -3X^2 - 3X + 1$.*

[1] In the Module-LWE notation, we keep n to define $\mathbb{Z}_q[X]/X^n + 1$

- $i = 4$: $\mathbf{a}_4 = (2X, 6X^2 + X + 2)$, $e_4 = X + 1$, $\langle \mathbf{a}_4, \mathbf{s} \rangle + e_4 = -3X^2 + 4X + 4$.
- $i = 5$: $\mathbf{a}_5 = (3X^2 + X, 2)$, $e_5 = X^2$, $\langle \mathbf{a}_5, \mathbf{s} \rangle + e_5 = 5X^2 + 3X - 5$.
- $i = 6$: $\mathbf{a}_6 = (3X + 1, X^2 - 1)$, $e_6 = X - 1$, $\langle \mathbf{a}_6, \mathbf{s} \rangle + e_6 = 2X^2 + 4X$.

From a matrix perspective, the challenge is given by

$$\mathbf{A} = \begin{pmatrix} X & X^2 \\ X^2 & X+1 \\ 5 & 3X^2+3 \\ 2X & 6X^2+X+2 \\ 3X^2+X & 2 \\ 3X+1 & X^2-1 \end{pmatrix} \in \left(\mathbb{Z}_{11}[X]/X^3+1\right)^{6\times 2}, \quad \mathbf{b} = \mathbf{A}\mathbf{s} = \begin{pmatrix} 1 \\ 2X^2+X-2 \\ -3X^2-3X+1 \\ -3X^2+4X+4 \\ 5X^2+3X-5 \\ 2X^2+4X \end{pmatrix}$$

18.3 Solving Module-LWE with Euclidean lattices

Here, we present one of the attaks against Module-LWE that uses Euclidean lattices. In addition to attacks using lattice methods, there is another connection between Module-LWE and the search for short vectors in lattices that we will know describe in this course: the *security reduction*. In a nutshell, there is a proof that if one solves Module-LWE, then one is capable of finding short vectors in certain Euclidean lattices. Since the latter is assumed to be a computationally difficult problem, this proof suggests that we should treat Module-LWE as a problem at least as hard as the search for short vectors. This security proof is outside of the scope of this course. Instead, we show that someone with access to an algorithm that can return the shortest non-zero vector of a Euclidean lattice can use it to mount an attack to solve Module-LWE.

We start by presenting this attack in the case of LWE (i.e. $n = 1$). We are given $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$, and $\mathbf{b} \in \mathbb{Z}_q^m$ with the promise that there is a rather short $\mathbf{e} \in \mathbb{Z}_q^n$ such that

$$\mathbf{A}\mathbf{s} + \mathbf{e} = \mathbf{b}.$$

Next, we observe that since $\mathbf{A}\mathbf{s} - \mathbf{b} = -\mathbf{e}$ is a short vector. This means that the vector

$$\begin{pmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & t \end{pmatrix} \begin{pmatrix} \mathbf{s} \\ -1 \end{pmatrix} = \begin{pmatrix} -\mathbf{e} \\ -t \end{pmatrix}$$

is also small (when the chosen parameter t is small). At this point, the matrix has coefficients in $\mathbb{Z}_q$. We can perform Gaussian elimination on the columns of A to obtain an invertible matrix $\mathbf{U} \in \mathbb{Z}_q^{n\times n}$ such that

$$\mathbf{A}\mathbf{U} = \begin{pmatrix} \mathbf{I}_{n\times n} \\ \mathbf{A}' \end{pmatrix}.$$

Note that if the upper $n \times n$ block of A is not invertible, Gaussian elimination will not yield the identity block, but this is unlikely to happen for large enough q. Additionally, in Lecture 14, we only formally defined Gaussian elimination on rows. The same process can happen with columns, where left multiplications are replaced by right multiplications. We do not re-define this process here. However, we can also easily reduce column Gaussian elimination to row Gaussian multiplication by finding $\mathbf{V}$ such that $\mathbf{V}\mathbf{A}^T = (\mathbf{I}_n \mid \mathbf{A'}^T)$, and notice that for $U = V^T$, we have $\mathbf{A}\mathbf{U} = \begin{pmatrix} \mathbf{I}_{n\times n} \\ \mathbf{A}' \end{pmatrix}$.

We can use this to simplify the search for $\mathbf{s}$ by performing a change of basis that replaces A by $\begin{pmatrix} \mathbf{I}_{n\times n} \\ \mathbf{A}' \end{pmatrix}$:

$$\begin{pmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & t \end{pmatrix} \begin{pmatrix} \mathbf{s} \\ -1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & t \end{pmatrix} \underbrace{\begin{pmatrix} \mathbf{U} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{U}^{-1} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix}}_{\text{Indentity}} \begin{pmatrix} \mathbf{s} \\ -1 \end{pmatrix} = \begin{pmatrix} \binom{\mathbf{I}_n}{\mathbf{A}'} & \mathbf{b} \\ \mathbf{0} & t \end{pmatrix} \begin{pmatrix} \mathbf{s}' \\ -1 \end{pmatrix}, \tag{18.1}$$

where $\mathbf{s}' = \mathbf{U}^{-1}\mathbf{s}$. Given the definition of $\mathbf{s}'$, we have that

$$\begin{pmatrix}\mathbf{I}_n \\ \mathbf{A}'\end{pmatrix}\mathbf{s}' - \mathbf{b} = \begin{pmatrix} s'_1 - b_1 \\ \vdots \\ s'_n - b_n \\ \mathbf{A}'\mathbf{s}' - (b_i)_{i>n}\end{pmatrix} = \mathbf{A}\mathbf{s} - \mathbf{b} = \mathbf{e} \pmod{q} \tag{18.2}$$

To turn the search for $\begin{pmatrix}\mathbf{s} \\ -1\end{pmatrix}$ into the search for a short vector in a Euclidean lattice, we need to work with matrices over $\mathbb{Z}$. In the following, we identify $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ with the matrix of $\mathbb{Z}^{m\times n}$ whose entries are the representatives of the entries of $\mathbf{A}$ in $\{\lceil -q/2\rfloor, \ldots, \lceil q/2\rfloor\}$ (where the rounding are such we have exactly q consecutive elements in the range). In terms of matrices in $\mathbb{Z}$, Equation (18.2) becomes

$$\begin{pmatrix}\mathbf{I}_n \\ \mathbf{A}'\end{pmatrix}\mathbf{s}' - \mathbf{b} = \begin{pmatrix} s'_1 - b_1 \\ \vdots \\ s'_n - b_n \\ \mathbf{A}'\mathbf{s}' - (b_i)_{i>n}\end{pmatrix} = \mathbf{e} + \begin{pmatrix} k_1 q \\ \vdots \\ k_m q\end{pmatrix} = \mathbf{e} + q\mathbf{I}_m\mathbf{k}$$

For some $\mathbf{k} \in \mathbb{Z}^m$. This means that once we handle integer matrices, we are searching for both $\mathbf{s}'$ and $\mathbf{k}$ such that

$$\begin{pmatrix}\mathbf{I}_n \\ \mathbf{A}'\end{pmatrix}\mathbf{s}' - \mathbf{b} + q\mathbf{I}_m\mathbf{k} = \begin{pmatrix} s'_1 - b_1 \\ \vdots \\ s'_n - b_n \\ \mathbf{A}'\mathbf{s}' - (b_i)_{i>n}\end{pmatrix} + q\mathbf{I}_m\mathbf{k}$$

is minimal. All vectors congruent to $\mathbf{s}'$ modulo q are solution (we just need to adjust the vector $\mathbf{k}$). We can therefore assume that the one we find is the one that ensures that $\forall i \leq n$, $s'_i - b_i \in \{\lceil -q/2\rfloor, \ldots, \lceil q/2\rfloor\}$. With this choice, we do not care about adjusting the rows of index $i = 1, \ldots, n$ by multiples of q because $s'_i - b_i$ are already reduced modulo q. We are therefore only trying to find $\mathbf{k} \in \mathbb{Z}^{m-n}$ that minimalizes

$$\begin{pmatrix}\mathbf{I}_n \\ \mathbf{A}'\end{pmatrix}\mathbf{s}' - \mathbf{b} + \begin{pmatrix}\mathbf{0}_n \\ q\mathbf{I}_{m-n}\mathbf{k}\end{pmatrix}$$

In terms of lattices, we want to find the shortest non-zero vector in the Euclidean lattice defined by the following matrix:

$$\mathbf{B} = \begin{pmatrix}\begin{pmatrix}\mathbf{I}_n \\ \mathbf{A}'\end{pmatrix} & \mathbf{b} & \begin{pmatrix}\mathbf{0}_n \\ q\mathbf{I}_{m-n}\end{pmatrix} \\ \mathbf{0} & t & 0\end{pmatrix}$$

With that choice, we have that $\mathbf{B}\begin{pmatrix}\mathbf{s}' \\ -1 \\ \mathbf{k}\end{pmatrix} = \begin{pmatrix}\mathbf{e} \\ -t\end{pmatrix}$ is the shortest vector of the lattice generated by the columns of $\mathbf{B}$ (assuming that $\mathbf{e}$ was short enough).

Example 3 *Let $n = 4$, $m = 7$, $q = 1009$ and*

$$\mathbf{A} = \begin{pmatrix} 208 & 598 & 985 & 275 \\ 168 & 34 & 519 & 353 \\ 478 & 86 & 389 & 53 \\ 732 & 928 & 54 & 991 \\ 817 & 906 & 347 & 259 \\ 858 & 442 & 513 & 936 \\ 671 & 805 & 277 & 984 \end{pmatrix} \quad \mathbf{e} = \begin{pmatrix} 1 \\ -1 \\ 1 \\ 2 \\ 0 \\ 1 \\ -2 \end{pmatrix} \quad \mathbf{s} = \begin{pmatrix} 99 \\ 61 \\ 424 \\ 516 \end{pmatrix} \quad \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} = \begin{pmatrix} 112 \\ 156 \\ 675 \\ 417 \\ 204 \\ 148 \\ 118 \end{pmatrix}$$

The Gaussian elimination on the column of $\mathbf{A}$ *yield*

$$\begin{pmatrix}\mathbf{I}_4\\ \mathbf{A}'\end{pmatrix} = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ 419 & 330 & 647 & 918\\ 587 & 413 & 102 & 387\\ 756 & 746 & 507 & 319\end{pmatrix}$$

Finally, the matrix $\mathbf{B}$ *is given by*

$$\mathbf{B} = \begin{pmatrix}\begin{pmatrix}\mathbf{I}_4\\ \mathbf{A}'\end{pmatrix} & \mathbf{b} & \begin{pmatrix}\mathbf{0}_{4\times 3}\\ q\mathbf{I}_3\end{pmatrix}\\ \mathbf{0} & 1 & 0\end{pmatrix} = \begin{pmatrix}1 & 0 & 0 & 0 & 112 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 156 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 675 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 417 & 0 & 0 & 0\\ 419 & 330 & 647 & 918 & 204 & 1009 & 0 & 0\\ 587 & 413 & 102 & 387 & 148 & 0 & 1009 & 0\\ 756 & 746 & 507 & 319 & 118 & 0 & 0 & 1009\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\end{pmatrix}$$

Lattice reduction on the lattice generated by the columns of $\mathbf{B}$ *yields the short vector*

$$\begin{pmatrix}1\\ -1\\ 1\\ 2\\ 0\\ 1\\ -2\\ 1\end{pmatrix} = \begin{pmatrix}\mathbf{e}\\ 1\end{pmatrix}$$

Under the assumption that $\pm\begin{pmatrix}\mathbf{e}\\ 1\end{pmatrix}$ *are the only short vectors, we know* $\pm\mathbf{e}$*, and we can recover* $\mathbf{As}$ *from* $\mathbf{As}+\mathbf{e}$*, and perform Gaussian elimination to recover* $\mathbf{s}$ *(see section "Learning Without Errors"or Lecture 14).*

Solving Module-LWE from Euclidean lattices can be done in a similar way. The trick is to use circulant matrices to turn multiplications in $\mathbb{Z}_q[X]/X^n+1$ into matrix-vector multiplications. As we increase the size of the matrices, it can be difficult to keep track of the operations. To illustrate this approach, let us take only rows 1 and 2 of Example 2. We have that $\mathbf{s} = (X+1, X-1)$. The matrix challenge is given by

$$\mathbf{A} = \begin{pmatrix}X & X^2\\ X^2 & X+1\end{pmatrix} \longleftrightarrow \begin{pmatrix}0 & 0 & -1 & 0 & -1 & 0\\ 1 & 0 & 0 & 0 & 0 & -1\\ 0 & 1 & 0 & 1 & 0 & 0\\ 0 & -1 & 0 & 1 & 0 & -1\\ 0 & 0 & -1 & 1 & 1 & 0\\ 1 & 0 & 0 & 0 & 1 & 1\end{pmatrix} = \mathbf{A}', \quad \mathbf{b} = \begin{pmatrix}1\\ 2X^2+X-2\end{pmatrix} \longleftrightarrow \begin{pmatrix}1\\ 0\\ 0\\ -2\\ 1\\ 2\end{pmatrix} = \mathbf{b}'$$

Hence the search for $\mathbf{s} \in \left(\mathbb{Z}_q[X]/X^3+1\right)^2$ and $\mathbf{e} \in \left(\mathbb{Z}_q[X]/X^3+1\right)^2$ such that $\mathbf{As}+\mathbf{e} = \mathbf{b}$ boils down to the search for $\mathbf{s}' \in \mathbb{Z}_q^6$ and $\mathbf{e}' \in \mathbb{Z}_q^6$ such that $\mathbf{A}'\mathbf{s}' + \mathbf{e}' + \mathbf{b}$, which can be done with the lattice technique described above.

18.4 A KEM from Module-LWE encryption

First, we describe how to do public-key encryption from Module-LWE, and then we recall how a KEM can be desined from this encryption scheme. We use the notation $R = \mathbb{Z}_q[X]/X^n + 1$, and we assume that we have probability distributions χ on R and χ^k on R^k (we did mention that it was better to specify them, but this section is about showing similarities with LWE encryption).

Key generation To generate the key, one performs the following steps:

- Draw $\mathbf{A} \in R^{k \times k}$.
- $\mathbf{s}, \mathbf{e} \leftarrow \chi^k$
- Pk $\leftarrow \mathbf{A}, \mathbf{As} + \mathbf{e}$, Sk $\leftarrow \mathbf{e}$.

Encryption We wish to encrypt a message $m \in \{0,1\}^n$ which is identified with the congruence class of $m_{n-1}X^{n-1} + \ldots + m_0$ in R. We use Pk $\leftarrow \mathbf{A}, \mathbf{As} + \mathbf{e}$. We perform the following steps:

- $\mathbf{r} \leftarrow \chi^k$, $\mathbf{e}_1 \leftarrow \chi^k$, $e_2 \leftarrow \chi$.
- $\mathbf{u} \leftarrow \mathbf{rA} + \mathbf{e}_1$,
- $v \leftarrow \mathbf{r}\,(\mathbf{As} + \mathbf{e}) + e_2 + \lceil \frac{q}{2} \rfloor \cdot m$.

The ciphertext is $c = (\mathbf{u}, v)$.

Decryption The ciphertext we receive is $c = (\mathbf{u}, v)$ with

$$\mathbf{u} = \sum_i r_i \mathbf{a}_i + \mathbf{e}_1$$
$$v = \left(\sum_i r_i \mathbf{a}_i\right) \mathbf{s} + \sum_i x_i e_i + e_2 + \left\lceil \frac{q}{2} \right\rfloor \cdot m.$$

We use the secret key $\mathbf{s}$ to perform the following operation:

$$v - \mathbf{us} = \underbrace{\sum_i x_i e_i + e_2 - \mathbf{e}_1 \mathbf{s}}_{e'} + \left\lceil \frac{q}{2} \right\rfloor \cdot m$$

As long as the coefficients of $e' \in R = \mathbb{Z}_q[X]/X^n + 1$ are less than $q/4$, we can recover each m_i by setting it to 0 if the coefficient i of $v - \mathbf{us}$ is less than $q/4$ and 1 otherwise.

Encapsulation Given Pk and a public hash function H, we can derive a key K and a ciphertext with the following operations:

- Draw $m \in \{0,1\}^n$ uniformly at random.
- $\hat{K} \leftarrow H(\text{Pk}||m)$.
- $c \leftarrow \text{Enc}(m, \text{Pk})$.
- $K \leftarrow H(\hat{K}||c)$.

Decapsulation Given Sk, a public hash function H and c, we retrieve K:

- $m \leftarrow \mathrm{Dec}(c, \mathrm{Pk})$.
- $\hat{K} \leftarrow H(\mathrm{Pk}||m)$.
- $K \leftarrow H(\hat{K}||c)$.