CIA - Le Futur et l'Horizon

KEMs et Calcul sur Données Chiffrées

Loïc Rouquette
EPITA // IF
9 octobre 2025

1 Introduction

Table des matières i

Introduction

KEM Post-Quantiques - Le Nouveau Standard d'Échange de Clés
- Un Nouveau Paradigme
- Étude de Cas : ML-KEM (CRYSTALS-Kyber)

Chiffrement Totalement Homomorphe (FHE) : Calculer dans un monde chiffré.
- Un Concept Révolutionnaire
- Les Défis Pratiques du FHE
- Panorama des Schémas FHE Modernes

Applications Concrètes et Vision d'Avenir

2

KEM Post-Quantiques - Le Nouveau Standard d'Échange de Clés

Objectif : Établir un Sécret Partagé

- Le but fondamental : permettre à deux parties (Alice et Bob) de s'accorder sur un secret partagé sur un **canal public**.
- Approche classique (vulnérable au quantique) : Échange de clés Diffie-Hellman.
- Approche Post-Quantique : **Mécanisme d'Encapsulation de Clé (KEM)**.

Définition Formelle d'un KEM

- Un KEM (*Key Encapsulation Mechanism*) est un triplet de 3 algorithmes :
  1. `KeyGen()` → `(pk, sk)` : Génère une paire de clés publique/privée.
  2. `Encaps(pk)` → `(ct, ss)` : Prend une clé publique, génère un **secret partagé** `ss` et un **chiffré** `ct` qui "encapsule" ce secret.
  3. `Decaps(sk, ct)` → `ss` : Prend une clé privée et un chiffré, et récupère le secret partagé original.

KEM vs. Diffie-Hellman : Analyse Comparative

Caractéristique	Protocole de Diffie-Hellman	Mécanisme d'Encapsulation de Clé (KEM)
Rôle	Symétrique et contributif	Asymétrique et unilatéral
Interactivité	Nécessairement interactif	Non-interactif (utile pour la com. asynchrone)
Source du Secret	Dérivé conjointement par calcul	Généré aléatoirement par une partie

Le Défi de la Confidentialité Persistante (PFS)

- **PFS** : La compromission d'une clé à long terme ne doit pas compromettre les sessions passées.
- Avec **Diffie-Hellman Éphémère (DHE)**, la PFS est une propriété inhérente. Les clés de session sont détruites, rendant le recalcul du secret impossible.
- Avec les **KEM**, la PFS n'est **pas inhérente**. Elle doit être assurée par le protocole (*e.g.* : TLS 1.3) en utilisant des clés KEM éphémères qui sont générées et détruites à chaque session.

Rôle dans le Chiffrement Hybride

- **Problème :** La cryptographie asymétrique est trop lente pour chiffrer de gros volumes de données.
- **Solution :** Le chiffrement hybride combine le meilleur des deux modes.
  - **KEM (Asymétrique) :** Établir efficacement un secret partagé `ss` de 256 bits.
  - **DEM (Symétrique) :** Utiliser `ss` comme clé pour un algorithme rapide (*i.e.* : AES-GCM) pour chiffrer/déchiffrer les données réelles.

ML-KEM : Le Standard du NIST

- Anciennement **CRYSTALS-Kyber**, formalisé dans la norme **FIPS 203**.
- Choisi par le NIST après un processus international pour son excellent équilibre sécurité/performance.

Bases Mathématiques : Le Problème LWE

- La sécurité repose sur la difficulté du **Learning With Errors (LWE)**.
- **Concept :** Résoudre un système d'équations linéaires légèrement "bruité".
- Étant donné A et t publics, trouver le secret s dans $t = A \cdot s + e$ est difficile à cause du petit vecteur d'erreur e. Sans e, ce serait trivial.

Optimisation : De LWE à Module-LWE

- Pour réduire la taille des clés publiques, on utilise des structures algébriques.
- On remplace les matrices de nombres par des **matrices de polynômes**.
- Les calculs sont effectués dans un anneau de polynômes : $R_q = \mathbb{Z}_q[X]/(X^n + 1)$, où les coefficients sont des entiers modulo q.

Échange Complet

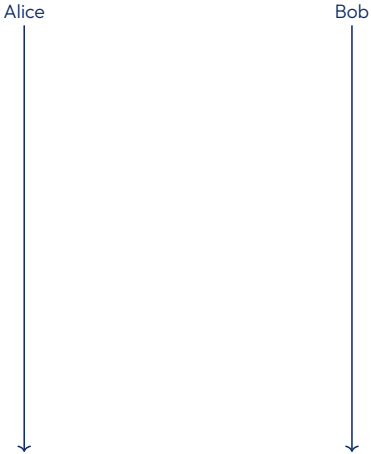

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

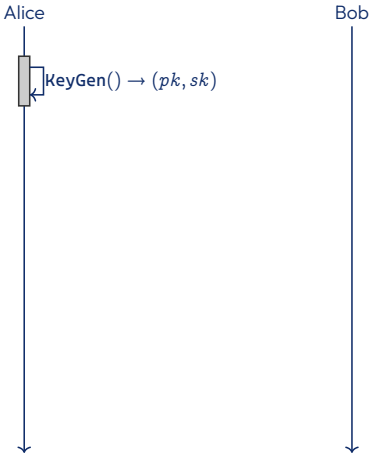

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

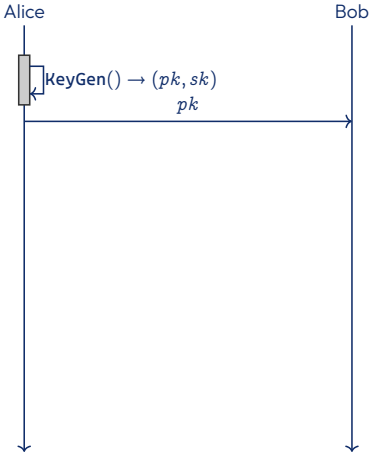

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

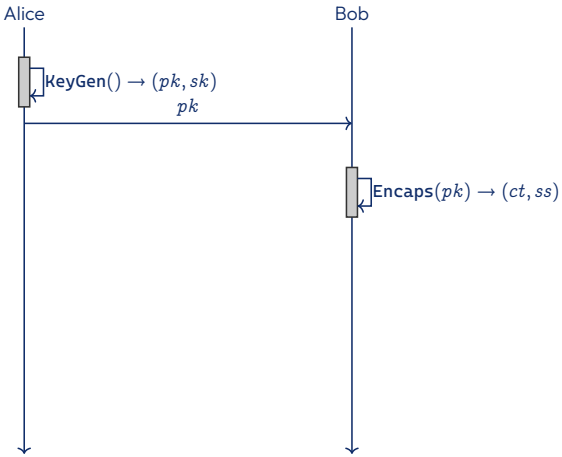

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

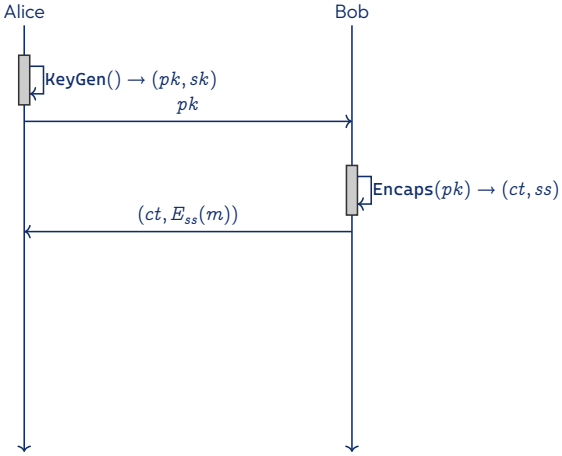

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

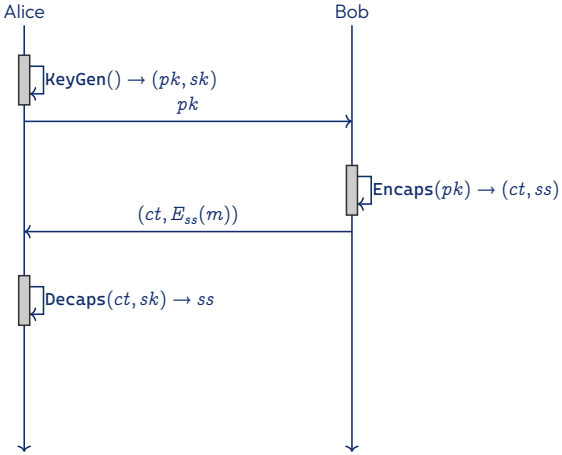

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Échange Complet

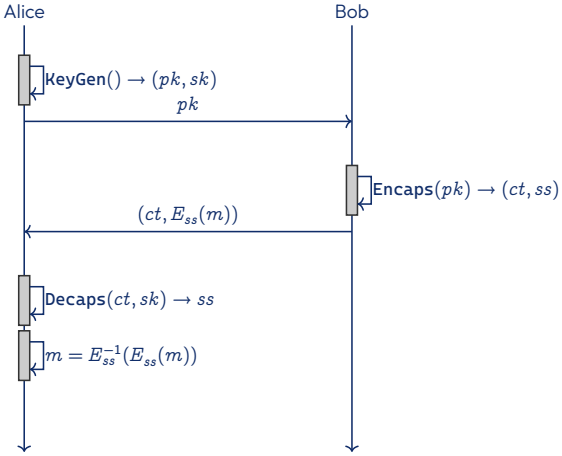

Figure 1 : Diagramme de séquence de l'utilisation d'un KEM

Algorithme Kyber : `KeyGen()`

Entrée

Aucune

Processus

1. Générer une matrice publique de polynômes A.
2. Générer un vecteur secret s et un vecteur de bruit e (avec de petits coefficients).
3. Calculer le vecteur public $t = A \cdot s + e$.

Sorties

Clé publique $pk = (A, t)$ et clé privée $sk = s$.

Algorithme Kyber : Encaps(pk)

Entrées

Clé publique pk

Processus

1. Générer un message aléatoire m.
2. Générer un vecteur aléatoire r et des bruits e_1, e_2 (petits coefficients).
3. Calculer le chiffré $ct = (u, v)$, où :
   - $u = A^T \cdot r + e_1$
   - $v = A^T \cdot r + e_2 + \texttt{Encode}(m)$

Sorties

Le Chiffré ct et le secret partagé $ss = \texttt{Hash}(m, ct)$.

Algorithme Kyber : `Decaps(sk, ct)`

Entrées

Clé privée sk et chiffré ct.

Processus

1. Calculer $v' = v - s^T \cdot u$
2. En substituant, la magie opère et les termes s'annulent.
3. On obtient $v' = \textbf{Encode}(m) + \text{bruit global}$.

La "Magie" de la Décapsulation

- $v' = (t^T + e_2 + \mathbf{Encode}(m)) - s^T(A^T r + e_1)$.
- Commet $t = A \cdot s + e$, on remplace t^T par $s^T A^T + e^T$.
- $v' = ((s^T A^T + e^T)r + e_2 + \mathbf{Encode}(m)) - s^T A^T r - s^T e_1$.
- Les termes $s^T A^T r$ s'annulent.
- Il reste : $v' = \mathbf{Encode}(m) + \underbrace{(e^T r + e_2 - s^T e_1)}_{\text{bruit global}}$.

Le Rôle Central du Bruit

- **Pour la sécurité :** Le bruit e dans la clé publique $t = A \cdot s + e$ masque le secret s, rendant le problème LWE difficile.
- **Pour la correction :** Le bruit global doit rester suffisamment petit pour ne pas corrompre le message lors du décodage. Le décodage arrondit simplement au coefficient le plus proche.

Aspects Pratiques : Sécurité et Performance

- **Sécurité :** La sécurité de Kyber est réductible au problème Module-LWE et vise la norme **IND-CCA2** (indistinguabilité sous attaque à chiffré choisi).
- **Performance :** Kyber est très rapide. Les multiplications de polynômes sont accélérées par la **Transformée en Nombre Théorique (NTT)**, un analogue de la FFT sur les coprs finis.

Paramètres de ML-KEM (FIPS 203)

Niveau de Sécurité	Équivalent	Taille de Clé Publique (octets)	Taille Chiffré (octets)
ML-KEM-512	AES-128	800	768
ML-KEM-768	AES-192	1184	1088
ML-KEM-1024	AES-256	1568	1568

Résumé de la Partie 1

- Les KEMs sont le nouveau paradigme non-interactif pour l'échange de clés PQC.
- ML-KEM (Kyber) est le standard du NIST, basé sur la difficulté mathématique du Module-LWE.
- Il repose sur un équilibre subtil entre le bruit (pour la sécurité) et la correction.
- C'est une technologie standardisée et prête pour le déploiement.

Questions?

Chiffrement Totalement Homomorphe (FHE) : Calculer dans un monde chiffré.

Au-delà de la Sécurité en Transit et au Repos

- Les KEM PQC sécurisent les données

Au-delà de la Sécurité en Transit et au Repos

- Les KEM PQC sécurisent les données **en transit**

Au-delà de la Sécurité en Transit et au Repos

- Les KEM PQC sécurisent les données **en transit** et **au repos**.

Au-delà de la Sécurité en Transit et au Repos

- Les KEM PQC sécurisent les données **en transit** et **au repos**.
- Le FHE s'attaque à la dernière frontière : la sécurisation des données **en cours d'utilisation**.

Le "Saint Graal" de la Cryptographie

- **Principe :** Effectuer des calculs arbitraires sur des données tout en les maintenant chiffrées.
- Un tiers (*e.g.* : un service cloud) peut manipuler des données sans jamais y avoir accès en clair.
-

Définition Formelle du FHE

- Un schéma de chiffrement est homomorphe si :

$$\texttt{Decrypt}(sk, \texttt{Eval}(pk, f, \text{Enc}(pk, m_1), \dots)) = f(m_1, \dots)$$

- En clair : déchiffrer le résultat d'un calcul sur les chiffrés est identique à effectuer le calcul sur les données claires.

Classification des Schémas Homomorphes

Classification des Schémas Homomorphes

- **Partiellement Homomorphe (PHE) :** Nombre illimité d'opération d'un seul type (addition OU multiplication).

Classification des Schémas Homomorphes

- **Partiellement Homomorphe (PHE) :** Nombre illimité d'opération d'un seul type (addition OU multiplication).
- **Quelque peu Homomorphe (SHE) :** Nombre *limité* d'additions ET de multiplications.

Classification des Schémas Homomorphes

- **Partiellement Homomorphe (PHE) :** Nombre illimité d'opération d'un seul type (addition OU multiplication).
- **Quelque peu Homomorphe (SHE) :** Nombre *limité* d'additions ET de multiplications.
- **Totalement Homomorphe (FHE) :** Nombre *illimité* d'additions ET de multiplications permettant d'évaluer n'importe quelle fonction.

Le Mécanisme Central : Le Bruit

- Comme les KEM basés sur LWE, les schémas FHE modernes introduisent du bruit dans chaque chiffré pour la sécurité.
- **Le problème :** Chaque opération homomorphe (surtout la multiplication) **augmente le bruit** dans le chiffré résultant.

Le "Budget de Bruit"

- Chaque chiffré possède un ”budget de bruit” intrinsèque.
- Si le bruit accumulé dépasse un certain seuil, le déchiffrement échoue et produit un résultat incorrect.
- C’est la raison pour laquelle un schéma SHE a une ”profondeur” de circuit limitée.

La Percée de Gentry (2009) : Le Bootstrapping

- C'est la procédure révolutionnaire qui transforme un schéma SHE en un schéma FHE.

La Percée de Gentry (2009) : Le Bootstrapping

- C'est la procédure révolutionnaire qui transforme un schéma SHE en un schéma FHE.
- **Ojectif :** "Réinitialiser" le bruit d'un chiffré pour lui redonner un budget de calcul.

Comment Fonctionne le Bootstrapping ?

- L'idée est d'utiliser le schéma SHE lui-même pour évaluer son propre circuit de déchiffrement de manière homomorphe.
- **Processus :**
  1. On dispose d'une version **chiffrée de la clé privée.**
  2. Pour un chiffré bruyant ct, on évalue homomorphiquement la fonction $\texttt{Decrypt}(\texttt{Enc}(pk', sk), ct)$.
  3. Le résultat est un **nouveau chiffré**, ct', qui contient le même message mais avec un niveau de bruit faible et "rafraîchi".

Défi 1 : La Performance

- Le bootstrapping, bien que puissant, est une opération **extrêmement coûteuse en calcul**.
- Les opérations homomorphes sont de plusieurs ordres de grandeur plus lentes que leurs équivalents en clair.
- C'est le principal obstacle à une adoption généralisée.

Défi 2 : L'Expansion des Chiffrés

- Les chiffrés FHE sont **beaucoup plus volumineux** que les données en clair qu'ils représentent.
- Le facteur d'expansion peut être de l'ordre de 1000x à 100 000x. Chiffrer un entier peut produire des dizaines de Ko de données.
- Impact majeur sur la bande passante et le stockage.

Défi 3 : La Complexité des Paramètres

- La mise en place d'un système FHE nécessite un choix minutieux de nombreux paramètres (taille de polynômes, modules, distribution de bruit, etc.).
- C'est un compromis constant entre **sécurité**, **correction** (budget de bruit suffisant) et **performance**.
- Il n'existe pas de solution "universellement optimale" ; le choix est spécifique à l'application.

Schémas pour l'Arithmétique Exacte (BFV/BGV)

Principe

Conçus pour les opérations exactes sur des entiers (arithmétique modulaire).

Gestion du bruit

Le bruit occupe les bits de poids faible et ne doit pas ”déborder sur le message”.

Cas d’usage

Vote électronique sécurisé, requêtes privées sur des bases de données, calculs statistiques exacts.

Schéma pour l'Arithmétique Approchée (CKKS)

Principe

Conçu pour les calculs sur des nombres réels ou complexes de manière approchée. Essentiel pour la data science.

Gestion du bruit

Le bruit cryptographique est traité comme faisant partie de l'erreur d'arrondi du calcul, ce qui est très efficace.

Cas d'usage

Le schéma de prédilection pour le **Machine Learning préservant la confidentialité (PPML)**.

Schéma pour les Circuits Booléens (TFHE)

Principe

Optimisé pour l'évaluation de portes logiques (AND, OR, NOT) sur des bits individuels chiffrés.

Innovation Clé

Un **bootstrapping extrêmement rapide** (quelques millisecondes) qui peut être executé après chaque porte logique pour maintenir un bruit bas en permanence.

Le Bootstrapping Programmable de TFHE

- Le boostrapping dans TFHE est ”programmable”.
- Il peut évaluer une fonction arbitraire à 1 variable (via une table de consultation) **en même temps qu'il rafraîchit le chiffré**, sans coût supplémentaire significatif.
- Ceci est extrêmement puissant pour évaluer des fonctions non-linéaires.

Table Comparatif des Schémas FHE

Schéma	Type d'Arithmétique	Bootstrapping	Application Principale
BFV/BGV	Entiers (exacte)	Lent	Vote, BDD
CKKS	Réels (approchée)	Modéré	Machine Learning
TFHE	Bits (exacte)	Très rapide & Programmable	Circuits, Smart Contracts

Résumé Partie 2

- Le FHE permet de calculer des données chiffrées, mais à un coût de performence significatif.
- La gestion du bruit est le défi central, résolu par le bootstrapping.
- Les schémas modernes (BFV, CKKS, TFHE) sont spécialisés pour différents types de calculs.
- La technologie est encore en phase de recherche active mais progresse très rapidement.

Questions?

4

Applications Concrètes et Vision d'Avenir

Application Phare : Le ML Préservant la Confidentialité (PPML)

- Le cas d'usage le plus mature est l'**inférence en tant que service (MLaaS)**.
- **Scénario :** Un client chiffre ses données (*e.g.* : une radio médicale), un serveur applique un modèle de détection de tumeur de manière homomorphe, et renvoie un diagnostic chiffré.

Défi du PPML : Les Fonctions d'Activation

- Les réseaux de neurones utilisent des fonctions non-linéaires comme **ReLU** ($\max(0, x)$).
- Le FHE ne support que les polynômes.
- **Solution :** Remplacer **ReLU** par des **approximations polynomiales** (*e.g.* : un polynôme quadratique). Cela introduit une erreur d'approximation qui doit être gérée.

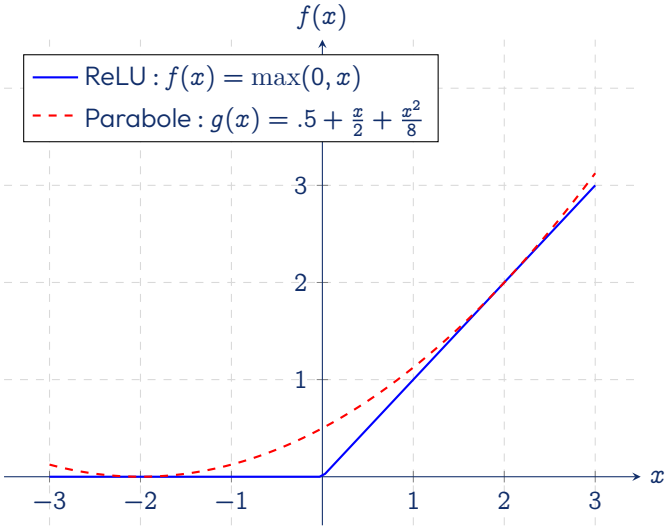

Figure 2 : Comparaison entre la fonction ReLU et une parabole d'approximation.

Outils et Compilateurs FHE

- La gestion manuelle de la profondeur du circuit et du placement des bootstrappings est très complexe.
- Des **compilateurs spécialisés** (*e.g.* : `Concrete-ML`, `Orion`) automatisent ce processus.
- Ils permettent de convertir des modèles entraînés avec des outils standards (`Scikit-Learn`) en leurs équivalents FHE exécutables.

Synergie 1 : FHE + Confidential Computing (TEE)

- Les TEE (Trusted Execution Environments) sont des enclaves matérielles sécurisées.
- **Complémentarité :** Un TEE peut être utilisé pour exécuter les opérations de bootstrapping de manière beaucoup plus rapide, en déchiffrant et rechiffrant les données à l'intérieur de l'enclave.
- Le FHE protège les données contre les attaques par canaux auxiliaires, une faiblesse potientielle des TEE.

Synergie 2 : FHE + Blockchain

- **Paradoxe de la Blockchain :** Elle repose sur la transparence, ce qui est incomptabile avec la confidentialité des données.
- **Solution FHE :** Permet la création de **contrats intelligents confidentiels**. L'état du contrat reste chiffré sur la chaîne, et les noeuds exécutent la logique du contrat de manière homomorphe.

L'Écosystèmes des PETs

- Le FHE fait partie d'un écosystème plus large de **Technologies de Préservation de la Confidentialités (PETs)**.
- Il sera de plus en plus combiné avec :
  - **Calcul Multipartite Sécurisé (MPC) :** Calculer une fonction sur des entrées privées de plusieurs parties.
  - **Preuves à Divulgation Nulle de Connaissance (ZKP) :** Prouver une affirmation sans révéler l'information sous-jacente.

Conclusion : Maturité et Adoption

- **KEM Post-Quantiques : Prêts pour le déploiement.** Le défi est l'ingénieurie de la migration.
- **FHE :** Encore en **recherche et développement**. Le principal obstacle reste la performance, mais les progrès sont extrêmement rapides, notamment grâce à l'accélération matérielle.

Vision d'Avenir

- La véritable nouvelle frontière est l'ingénierie de **système cryptographiques hybrides**, où chaque technologie est utilisée pour ce qu'elle fait de mieux.
- Les KEM pour le transport, le FHE pour le calcul, les ZKP pour la vérification : les briques d'un avenir numérique plus intelligent et plus sûr.

Questions?