Réf : T0-229-295-316

Université Chouaib Doukkali
Ecole Nationale des Sciences Appliquées d'El Jadida
Département Télécommunications, Réseaux et Informatique

PROJET DE FIN D'ANNEE

Filière : **ISIC**

Niveau :**1ère Année**

Sujet :

Application du chiffrement homomorphe à la protection de la vie privée.

Réalisé Par :
MANARI Wissal
FOUZI Loubna
DALI Ali

Encadré par :
Pr. Ali KARTIT

Présenté le 20**/05/2024** devant le jury composé de :

Prof. Ali **KARTIT**, Professeur à l'ENSAJ
Prof . Noureddine **ASSAD** , Professeur à l'ENSAJ

Année Universitaire : 2023/2024

Résumé

Dans ce travail, nous avons utilisé une primitive cryptographique permettant de réaliser des opérations arithmétiques sur des données chiffrées. Grâce à celle-ci il est possible de confier des calculs à un agent externe, sans que les données traitées ou les résultats obtenus ne soient accessibles à cet agent. Ainsi il est possible de réaliser des protocoles améliorant la protection de la vie privée dans de nombreux domaines. La découverte de schémas de chiffrement homomorphe relativement efficaces, en général fondés sur les réseaux euclidiens, permet d'envisager des applications intéressantes à un cout calculatoire acceptable.

Un domaine d'application primordial de la protection de la vie privée est celui des données génomiques. Il est crucial de protéger ces données puisqu'elles sont liées aux individus et soulèvent des problèmes de vie privée qui sont difficiles à contourner. Nous proposons un protocole permettant d'externaliser les données génomiques chiffrées vers un serveur tout en ayant la possibilité de tester la présence d'un élément (une mutation) et ce de manière privée. Le protocole est base sur un retrait d'information privée et est efficace d'un point de vue consommation mémoire et performance temporelle. Les techniques utilisées permettent une protection de la vie privée optimale.

En fin, nous avons mis en place un algorithme permettant de détecter une fraude lors d'un paiement en ligne en utilisant le calcul homomorphe. Cet algorithme permet de prédire si un paiement est potentiellement frauduleux sans apprendre les informations de ce paiement.

Remerciements

Au moment de conclure ce projet de fin d'année, nous tenons à exprimer notre profonde gratitude pour votre soutien, votre guidance et votre expertise tout au long de cette aventure. Votre engagement et votre dévouement ont été une source d'inspiration pour nous, et nous avons énormément apprécié chaque instant passé à travailler sous votre direction. Mr. Ali **KARTIT** et Mr. Noureddine **ASSAD** .

Votre expertise a illuminé notre chemin et nous a permis d'accomplir des réalisations dont nous sommes fiers aujourd'hui. Votre patience, vos conseils avisés et votre encouragement ont été essentiels pour surmonter les défis rencontrés et pour nous aider à atteindre nos objectifs.

Votre investissement personnel dans notre réussite ne passe pas inaperçu. Vous avez été bien plus que des encadrants pour nous ; vous avez été des mentors bienveillants qui ont façonné notre expérience d'apprentissage de manière significative.

Nous sommes reconnaissants pour cette opportunité de travailler avec vous et de bénéficier de votre expertise. Les leçons que nous avons apprises sous votre direction resteront gravées en nous bien au-delà de ce projet.

Nous tenons à vous remercier sincèrement pour votre contribution précieuse à notre réussite. Votre influence positive continuera d'inspirer nos efforts futurs.

Avec toute notre gratitude.

Table des matières

Liste des figures

1
Introduction

La protection de la vie privée est devenue une préoccupation majeure dans un monde de plus en plus numérisé, où les données personnelles sont devenues une monnaie d'échange courante. Dans ce contexte, les avancées en cryptographie deviennent essentielles pour garantir la confidentialité des informations sensibles tout en permettant leur utilisation à des fins légitimes. Parmi ces avancées, le chiffrement homomorphe émerge comme une solution prometteuse pour permettre des calculs sur des données chiffrées sans compromettre leur confidentialité, Il présente une primitive cryptographique révolutionnaire, permet d'effectuer des opérations arithmétiques sur des données chiffrées sans avoir besoin de les déchiffrer au préalable. Cette capacité ouvre la voie à de nombreuses applications innovantes dans le domaine de la protection de la vie privée. En effet, elle permet de réaliser des calculs sur des données sensibles tout en préservant leur confidentialité, offrant ainsi de nouvelles perspectives dans divers domaines, tels que la santé, et les transactions financières.

Ce travail s'articule sur trois chapitres principaux :

Dans le premier chapitre, nous commencerons par une revue de l'état de l'art de la cryptographie, en examinant les concepts de base tels que la cryptographie asymétrique et symétrique, ainsi que les notions d'indistingabilité. Cette exploration approfondie jettera les bases nécessaires pour comprendre le chiffrement homomorphe et son importance dans le domaine de la protection de la vie privée.

Le deuxième chapitre sera consacré à l'examen des protocoles spécifiques basés sur le chiffrement homomorphe. Nous étudierons en détail des protocoles tels que le Private Information Retrieval (PIR) et l'Oblivious Transfer (OT), en examinant comment ils peuvent être utilisés pour garantir la confidentialité des données dans des scénarios réels.

Enfin, dans le troisième chapitre, nous explorerons les différentes applications du chiffrement homomorphe dans des domaines sensibles tels que la sécurité des données génomiques et la détection de la fraude lors des paiements en ligne. Nous analyserons comment cette technologie peut être mise en œuvre pour sécuriser les données sensibles et préserver la vie privée des individus dans un monde numérique en constante évolution.

2

Etat de l’art

Introduction :

La progression des cryptosystèmes vers des solutions plus avancées est intrinsèquement liée à l'importance grandissante de préserver la vie privée et la confidentialité des données dans un monde numérique en perpétuelle expansion. Le chiffrement homomorphe est une avancée importante dans ce domaine, car il permet de réaliser des opérations sur des données chiffrées sans les déchiffrer au préalable. Dans cette partie dédiée à l'état de l'art, nous examinerons les concepts essentiels du chiffrement homomorphe, en mettant l'accent sur deux problèmes majeurs : le problème LWE (Learning With Errors) et le problème RLWE (Ring Learning With Errors), ainsi que sur les méthodes de batching/bootstrapping et les diverses formes de chiffrement homomorphe.

2.1 Historique de la Cryptographie :

Le terme cryptographie vient des mots grecs kruptos << caché >> et graphein << écrire>> . Un des premiers documents connus recelant un secret enfoui dans l´écriture est une tablette en argile réalisée par un potier qui y avait dissimulé sa recette de fabrication en jouant sur l’orthographe des mots et en supprimant des consonnes. Elle a été retrouvée en Irak, et date du XVIème siècle avant Jésus Christ.

La cryptographie a pour but de s´sécuriser un message pour lequel on veut assurer des propriétés telles que la confidentialité (un adversaire ne peut pas lire un message) ou l’intégrité (un adversaire ne peut pas créer ou modifier un message valide). Parmi les algorithmes cryptographiques on peut distinguer les algorithmes de chiffrement, qui permettent d’assurer la confidentialité d’une donnée en combinant un message et une clé pour rendre le résultat (appelé chiffré) incompréhensible. Les premiers algorithmes de chiffrement ´étaient des algorithmes qui réalisaient des substitutions. Historiquement, on peut discerner trois types de chiffrement par substitution : mono-alphabétique (une lettre remplace une autre lettre), poly-alphabétique (une lettre est remplacée par une autre en fonction d’un ´état interne a` l’algorithme) et poly-gramme (un groupe de lettres remplace un autre groupe de lettres).

Le chiffré de César est un système simple par substitution mono-alphabétique consistant à décaler les lettres de l’alphabet. Le chiffrement est limité par le nombre de lettres de l’alphabet. Il est assez facile de retrouver le message initial en essayant tous les d´décalages.

Blaise de Vigenère présenta, en 1586, une technique de chiffrement par substitution poly-alphabétique. Ce chiffrement utilise une clé qui permet de remplacer chaque caractère. Plus la clé est grande et diversifiée, plus le chiffrement est solide. Des passages entiers de certaines œuvres littéraires pouvaient être utilisés. Ce chiffrement n'a été cassé qu'au milieu du XIXème siècle.

La cryptographie, par le passé, a souvent eu un rôle important lors des guerres. Le chiffrement de César a été par exemple utilisé par l'armée soviétique lors de la première guerre mondiale. Les connaissances en cryptographie des alliés lors de la première guerre mondiale seraient un élément de victoire essentiel. C'est dans ce contexte qu'est née Enigma, utilisée par les allemands lors de la seconde guerre mondiale. Il s'agit d'une machine portable utilisant des rotors sur cylindres afin de chiffrer et déchiffrer des messages. Il s'agit d'un chiffrement poly-gramme. Le chiffrement d'Enigma a été cassé par une ´équipe composée entre autres d'Alan Turing, qui est considéré comme le père de l'informatique.

2.2 Cryptographie asymétrique

En cryptographie asymétrique, Alice possède une paire de clés, une publique, l'autre privée. Tout le monde peut envoyer des messages à Alice en les chiffrant avec sa clé publique, seule Alice peut les déchiffrer en utilisant sa clé privée.

Définition 3.1.2. Un système de chiffrement asymétrique consiste en trois algorithmes :

1. KeyGen : prend en entrée un paramètre de sécurité λ, retourne une paire de clés (pk,sk) où pk désigne la clé publique utilisée pour chiffrer les messages et sk celle pour déchiffrer

2. Enc : prend en entrée le message m à chiffrer et la clé publique pk, retourne un message chiffré c

3. Dec : prend en entrée un message chiffré c et la clé privée sk, retourne le message déchiffré m

Dans la pratique, la gestion des clés publiques s'effectue par l'intermédiaire d'une infrastructure nommée PKI (Public Key Infrastructure), mais nous ne rentrerons pas dans les détails sur ce point.

2.3 Cryptographie symétrique

Contrairement aux cryptosystèmes asymétriques, dans les symétriques ou encore, à clés secrètes, Bob et Alice doivent posséder la même clé k pour pouvoir dialoguer ensemble.

Définition 3.1.3. Un cryptosystème symétrique consiste en deux algorithmes :

1. Enc : prend en entrée le message m à chiffrer et la clé secrète k, retourne le chiffré c de m

2. Dec : prend en entrée un message chiffré c et la clé secrète k, retourne le message déchiffré m

Bien que ces cryptosystèmes soient généralement beaucoup plus rapides, les utilisateurs doivent partager un secret. Le problème réside dans le processus d'échange des clés ainsi que le nombre de clés à posséder pour discuter avec plusieurs utilisateurs.

2.4 Indistingabilité

Une propriété importante pour un chiffré est l'indistingabilité . Un chiffrement possédant cette propriété permet d'envoyer des chiffrés sans que l'adversaire puisse apprendre d'information sur ce qui a été chiffré. Notamment un adversaire ne peut pas savoir si des chiffrés correspondent au même clair ou à des clairs différents.

2.4.1 Définition 1 (Fonction négligeable)

Une fonction négligeable est une fonction qui minore asymptotiquement l'inverse de tout polynôme.

2.4.2 Définition 2 (Indistingabilité face à une attaque à clairs choisis).

Considérons le jeu suivant :

Le challengeur génère (sk, pk) pour un paramètre λ et donne pk `a l'adversaire.

L'adversaire fait des opérations de son choix et soumet deux clairs distincts m_0 et m_1 au challengeur.

Le challengeur s´sélectionne de manière uniformément aléatoire un bit b et envoie le challenge c = Chiffrement(pk, mb).

L'adversaire fait des opérations de son choix et renvoie un bit b^0.

L'adversaire gagne si $b = b^0$

Un cryptosystème est dit indistingable pour une attaque `a clairs choisis si tout adversaire faisant un nombre polynomial (en λ) d'opérations a un avantage n´négligeable (en λ). Un adversaire a un avantage n´négligeable si sa probabilité de gagner le jeu ne dépasse pas

$\frac{1}{2} + \epsilon(\boldsymbol{\lambda})$ où ϵ est une fonction n´négligeable.

2.5 Chiffrement homomorphe

2.5.1 Définition

Le chiffrement homomorphe est une technique avancée en cryptographie qui permet de réaliser des opérations sur des données chiffrées sans avoir besoin de les déchiffrer au préalable. En d'autres termes, cela signifie que des calculs peuvent être effectués sur les données cryptées, et

les résultats de ces calculs, lorsqu'ils sont déchiffrés, correspondent aux résultats que l'on obtiendrait en effectuant les mêmes opérations sur les données en clair.

Cette capacité est cruciale dans le domaine de la protection des données sensibles, car elle permet d'effectuer des analyses et des traitements sur des informations confidentielles tout en maintenant leur confidentialité. Par exemple, dans un contexte où des données médicales doivent être analysées de manière sécurisée, le chiffrement homomorphe permettrait à des chercheurs d'effectuer des calculs sur les données chiffrées sans avoir accès aux informations médicales elles-mêmes.

Il peut être utilisé dans divers domaines, tels que la sécurité des données personnelles et médicales, les transactions financières sécurisées, et d'autres applications où la confidentialité des données est primordiale. Bien que complexe à mettre en œuvre, le chiffrement homomorphe offre un niveau de sécurité élevé tout en permettant des analyses et des traitements sur des données sensibles.

Étant donné $\boldsymbol{C}$ l'espace des chiffrés, $\boldsymbol{M}$ l'espace des messages, et $\boldsymbol{F}$ l'espace des fonctions évaluables, on définit un chiffrement homomorphe sur $\boldsymbol{F}$ le quadruplet d'algorithmes $(\boldsymbol{\kappa}, \boldsymbol{\varepsilon}, \boldsymbol{D}, \boldsymbol{Eval})$ tel que :

. $\boldsymbol{\kappa}$ **:** $\mathbb{N} \rightarrow \boldsymbol{PK} \times \boldsymbol{SK}$: la fonction de génération des clefs ;

. $\boldsymbol{\varepsilon}$: $\boldsymbol{M} \rightarrow \boldsymbol{C}$ est la fonction de chiffrement ;

. $\boldsymbol{D}$: $\boldsymbol{C} \rightarrow \boldsymbol{M}$ est la fonction de déchiffrement ;

. **EVAL** : F × C × C → C est la fonction d'évaluation .

On souhaite avoir les propriétés suivantes :

. La correction : D(ε(m)) = m ;

. L'homomorphisme :

∀ f ∈ F, D(Eval (f, C1, C2)) = f (D (C1), D (C2)

Pour des raisons de simplicité de lecture, on a décrit f comme une fonction d'arité 2, mais F peut décrire un espace de fonction d'arité quelconque.

2.5.2 Rappel mathématique

Dans cette partie, nous exposons les concepts et les caractéristiques fondamentales des structures algébriques, des polynômes et des réseaux euclidiens.

Un corps est un ensemble d'éléments et de deux opérations sur cet ensemble qui répondent à certaines relations. Cette structure est similaire à notre compréhension des nombres et des opérations que nous effectuons. Peut les mettre en œuvre. Avant de décrire les liens entre les deux opérations de la structure de corps, il est important de rappeler la structure de groupe et d'anneau, car un corps est un cas spécifique de la structure d'anneau, et cette dernière est plus riche que la structure du groupe.

Les Anneaux :

Un anneau A est ensemble muni d'une opération d'addition et d'une opération de multiplication $A \times A \rightarrow A$, telles que :

(A, +) est un groupe abélien, avec l'élément neutre noté $0 \in A$ et appelé zéro de l'anneau

L'opération × est associative, i.e., pour tous éléments a, b et c de A, $a \times (b \times c) = (a \times b) \times c$,

La multiplication × est distributive par rapport à +,i.e. pour tout élément a, b et c ;

$a \times (b+ c) = a \times b+ a \times c$ et $(a + b) \times c = (a \times c) + (b \times c)$.

Un anneau (A, +, ×) est dit commutatif si sa multiplication est commutative.

Un anneau (A, +, ×) s'appelle unitaire si l'opération × admet un élément neutre noté $1 \in A$ etappeléunité de l'anneau.

Polynômes :

Un polynôme P, en algèbre générale à une indéterminée sur un anneau unitaire est une expression de la forme :

$$P(X) = \sum_{i=0}^{n}(a_i.X^i) = a_0 + a_1.X^1 + a_2.X^2 + ... + a_n.X^n$$

où X est un symbole appelé « indéterminée du polynôme », supposé être distinct de tout élément de l'anneau, les coefficients ai son dans l'anneau noté K et n est un entier naturel.

Si tous les coefficients du polynôme P sont nuls, on dit que P est le polynôme nul et on le note 0. On appelle monôme un polynôme dont tous les coefficients sont nuls sauf, au plus, l'un d'entre eux.

On appelle degré de P, et on note deg(P), le plus grand entier $n \geq 0$ tel que $an \neq 0$.

On notera K[X] l'ensemble des polynômes à une indéterminée à coefficients dans l'anneau commutatif K, autrement dit, K[X] est l'anneau de polynômes à coefficients dans K.

Soient $P = \sum_{i=0}^{n}(a_i.X^i)$ et $Q = \sum_{i=0}^{n}(b_i.X^i)$ deux polynômes de K[X], On appelle de P et Q, et on note :

$P + Q = \sum_{i=0}^{n}(a_i + b_i)\ X^i$.

Le produit de P et Q est défini par :

P.Q = $\sum_{i=0}^{2n}(\sum_{k=0}^{i} a_k.b_{i-k}).X$ Où $a_k = 0$ et $b_k = 0$ pour tout $k \geq n+1$

On dit qu'un polynôme P de K[X] est premier ou irréductible sur le corps K s'il n'est pas constant et si ses seuls diviseurs dans K[X] sont les polynômes associés à P et les éléments non nuls de K.

Soit P∈K[X] un polynôme irréductible, alors K[X]/(P) est un corps. Les éléments de ce corps sont les polynômes premiers avec P, de degré inférieur à celui de P.

2.5.3 Réseaux euclidiens

Intuitivement, les réseaux euclidiens sont des arrangements réguliers de points dans l'espace euclidien $\mathbb{R}^n$. L'exemple le plus simple est $\mathbb{Z}^n$ formé par tous les points avec des coefficients entiers, mais tous les sous-groupes de $\mathbb{Z}^n$ sont aussi des réseaux. Une définition possible d'un réseau euclidien est la suivante.

Définition :

Un réseau euclidien L est un sous-groupe additif discret non vide de $\mathbb{R}n$ ($n \geq 0$).

Si le réseau n'est pas trivial, alors il est de cardinalité infinie et forme un module de type fini sur $\mathbb{Z}$. Ainsi, tout élément peut s'écrire comme la combinaison linéaire à coefficients entiers d'une certaine base. La proposition suivante caractérise les réseaux et introduit la notion de base d'un réseau.

Proposition 1 :

Soit L un réseau euclidien. Il existe des vecteurs (b_1, b_2, ..., b_d) linéairement indépendants de Rn tels que L est l'ensemble des combinaisons linéaires à coefficients entiers de (b_1, b_2, ..., b_d). Autrement dit, L est de la forme L : = $\{x \in \mathbb{R}^n ; x = \sum_{i=1}^{d} x_i b_i ; x_i \in \mathbb{Z}\}$.

Nous dirons que B : = (b_1, b_2, ..., b_d) forme une base du réseau L.

Toutes les bases d'un réseau L ont le même nombre d'éléments d appelé rang ou dimension du réseau. Souvent, un réseau L est représenté par une de ses bases B écrite sous forme matricielle. Dans toute cette thèse nous adoptons la convention suivante : les lignes de la matrice sont les coordonnées des vecteurs de la base. Ainsi avec un réseau de rang d, la matrice obtenue contient d lignes et n colonnes et appartient à $\mathbb{R}^{d \times n}$. Cette représentation n'est pas unique, mais elle est finie. La proposition qui suit montre le lien entre deux bases d'un même réseau.

Proposition 2 :

Soit (b1, b2, ..., bd) et (b'_1, b'_2, ..., b'_d) deux bases de L et B et B' les matrices (d ◊ n) respectives dans la base canonique de Rn. Alors la matrice de changement de base est une matrice inversible unimodulaire U (matrice avec des coefficients entiers et un déterminant qui vaut ±1) vérifiant B' = UB.

La proposition précédente montre que deux bases d'un réseau sont liées par une matrice unimodulaire. En fait, si B est une base et U une matrice unimodulaire, alors UB est aussi une base du réseau. Puisqu'il existe une infinité de matrices unimodulaires de dimension $d \geq 2$, un réseau L possède une infinité de bases. Pour tester si deux bases engendrent le même réseau, il suffit de calculer la matrice de passage d'une base à l'autre et de vérifier si elle est unimodulaire.

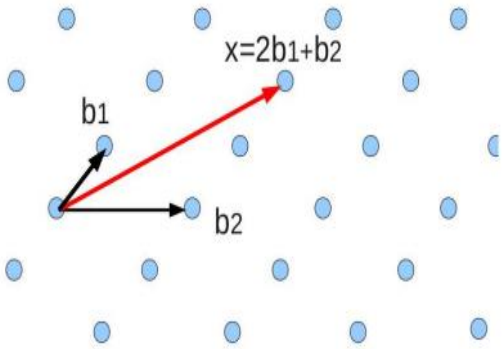


Figure 1 : Un réseau de dimension 2 et un vecteur x représenté sous la forme d'une combinaison linéaires des vecteurs (b1, b2).

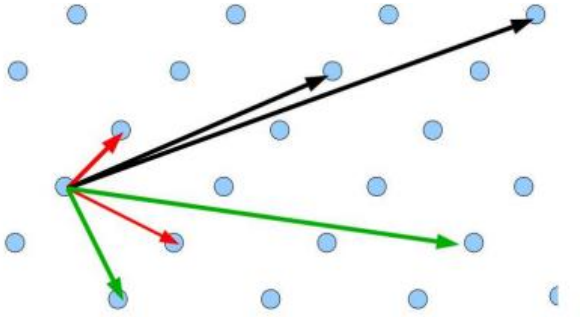


Figure 2: Un réseau de dimension 2 et trois bases équivalentes.

Orthogonalisation de Gram-Schmidt :

Une des propriétés d'une « bonne base » est son orthogonalité. Malheureusement, un réseau L ne possède pas forcement de base orthogonale. Pour mesurer l'orthogonalité d'une base, il est utile d'avoir une base orthogonale de référence, comme la base orthogonale de Gram-Schmidt**.**

Définition :

(Orthogonalisation de Gram-Schmidt) Soit (b_1, b_2,..., b_d) des <vecteurs linéairement indépendants. La base orthogonalisée de Gram-Schmidt de B = (b_1, b_2,..., b_d) est la famille de vecteurs linéairement indépendants $B^\star$= ($b_1^\star$, $b_2^\star$,..., $b_d^\star$) définie par :

$$b_1^\star = b_1 \ , \ b_i^\star = b\text{-}i \sum_{j=1}^{i} m \ m_{i,j} \ .b_j^\star \ \text{ pour } i \geq 2, \ \text{où} \quad m_{i,j} = \frac{\langle \mathbf{b}_i | \mathbf{b}_j^\star \rangle}{\|\mathbf{b}_j^\star\|^2}$$

On note P la matrice de passage de B à $B^\star$ ($B = PB^\star$) et la quantité $l_i := \| b_i^\star \|$ désigne la norme du i -ème vecteur orthogonalisé, appelée encore longueur de Siegel. La matrice de passage est une matrice triangulaire inférieure avec des 1 sur la diagonale et est donnée par :

La figure suivante représente une base orthogonale de Gram-Schmidt en dimension 2.

$$\mathcal{P} := \begin{array}{c} \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_i \\ \mathbf{b}_{i+1} \\ \vdots \\ \mathbf{b}_d \end{array} \begin{array}{c} \begin{array}{cccccccc} \mathbf{b}_1^\star & \mathbf{b}_2^\star & \dots & \mathbf{b}_{i-1}^\star & \mathbf{b}_i^\star & \mathbf{b}_{i+1}^\star & \dots & \mathbf{b}_d^\star \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ m_{2,1} & 1 & \dots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{i,1} & m_{i,2} & \dots & m_{i,i-1} & 1 & 0 & 0 & 0 \\ m_{i+1,1} & m_{i+1,2} & \dots & m_{i+1,i-1} & m_{i+1,i} & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{d,1} & m_{d,2} & \dots & m_{d,i-1} & m_{d,i} & m_{d,i+1} & \dots & 1 \end{pmatrix} \end{array}$$

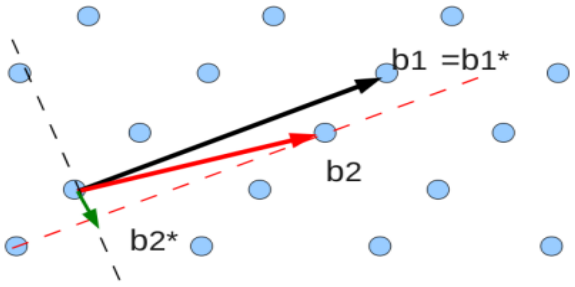


Figure 3: Deux vecteurs et leur orthogonalisation de Gram-Schmidt

Les vecteurs de $B^\star$ sont par construction des vecteurs orthogonaux et pour i = 1 ...d, les vecteurs ($b_1^\star$,..., $b_i^\star$) engendrent le même espace vectoriel que les vecteurs (b_1,..., b_i). Cependant, $B^\star$ n'est généralement pas une base du réseau. Pour le réseau donné par la Figure 3 engendré par les vecteurs (b1, b2), la base orthogonale ($b_1^\star$, $b_2^\star$) n'est pas une base du réseau. En outre, l'orthogonalisation de Gram-Schmidt dépend de l'ordre des vecteurs de la base B. Cette propriété est fondamentale pour l'algorithme **LLL** qui utilise des échanges entre deux vecteurs successifs de B pour "améliorer" son orthogonalité. La figure 4 montre l'effet d'un échange sur la base orthogonale B. On voit que la norme du vecteur $b_2^\star$augmente (ou diminue selon le sens de l'échange) ce qui donne une base plus ou moins « plate ».

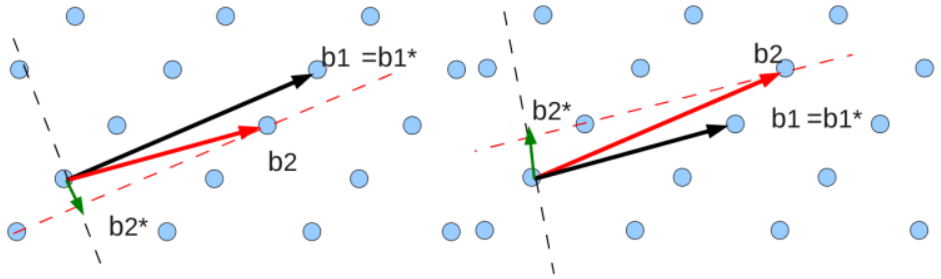


Figure 4 : Echange de deux vecteurs .

Du fait du lien entre la base B et son orthogonalisé de Gram-Schmidt, il est possible de retrouver plusieurs propriétés de la base B à partir de $B^\star$. La proposition suivante fait le lien entre un plus petit vecteur non-nul de B et les longueurs des vecteurs de la base orthogonale.

Proposition :

Soit un réseau L engendré par une base B. Soit $B^\star$ la base orthogonalisée de Gram Schmidt de la base B et v ∈ L(B) / {0} alors : $\| v \| \geq \min_{i \in [1..d]} \| b_i^\star \|$.

2.5.4 Les problèmes

Factorisation :

Tout nombre supérieur à 2 peut s'écrire sous-produit de facteurs premiers. Cependant, étant donné un nombre, il est généralement difficile de trouver ces facteurs premiers. Le meilleur algorithme (non quantique) permettant de factoriser est le crible algébrique sur les corps de nombres, dont la complexité est Ln [1 3, 3 q 64 9] = expo ((3 q 64 9 + o (1)) (ln(n))1/3

(ln(ln(n)))2/3. Le reste des problèmes disposent d'une version décisionnelle et d'une version calculatoire. Dans la version décisionnelle, l'attaquant a généralement pour but de distinguer une distribution d'une autre, alors que dans la version calculatoire il doit trouver une solution au problème.

Problème RSA :

Le problème RSA correspond à l'extraction d'une racine e-ième modulo un entier N = pq. Actuellement, la meilleure façon de procéder est de factoriser le module N. Formellement, étant donnés N = pq, c = me mod N, et e > 1, trouver m dans la version calculatoire, décider si un tel m existe dans la version décisionnelle.

Résidualité quadratique :

Le problème de résidualité quadratique peut être vu comme une instance du problème RSA où e = 2. Dans la version décisionnelle, le but est de déterminer si c est un carré modulo N = pq, dans la version calculatoire, l'attaquant doit trouver x tel que x2 = c mod N.

Problème de résidualité composée :

Le problème de résidualité composée consiste à décider si c ∈ ZN2 est une puissance Nième modulo N2 , où N = pq, et de trouver ses racines dans la version décisionnelle.

Problème de résidualité d'ordre supérieur :

Ce problème est une généralisation du problème de résidualité quadratique. Étant donnés des entiers N = pq et d|p - 1, déterminer si x ∈ Zn est une puissance d-ième modulo N. Formellement : déterminer si x = α d mod N, et trouver α dans la version calculatoire.

Logarithme discret :

Étant donnés un groupe cyclique G, un générateur g de ce groupe, et h = g x ∈ G, le problème du logarithme discret consiste à trouver x. Actuellement, il n'existe pas d'algorithme efficace permettant de résoudre ce problème. Cependant, si G = ZN avec N = p × q, pgcd (p, q) = 1, et que p et q sont connus, il est possible de faire mieux que le calcul naïf des puissances successives de g jusqu'à trouver h.

Problème de Diffie-Hellman :

Le protocole de Diffie Hellman est célèbre pour la solution qu'il apporte au problème d'échange des clés dans le domaine du chiffrement symétrique.

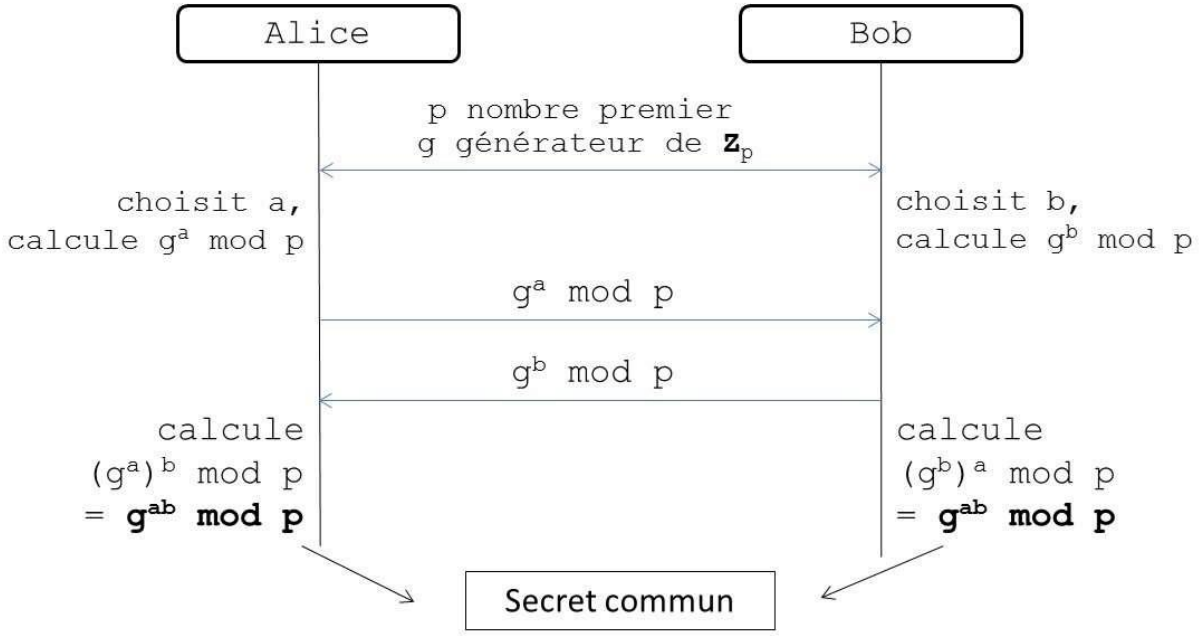


Figure 5 : Protocole d'échange de secret de Diffie - Hellman, 1976

Le problème de Diffie Hellman consiste à distinguer les distributions (g_a, g_b, g_{ab}) et (g_a, g_b, r) modulo le nombre premier p (pour r aléatoire), et à calculer g ab à partir de g a et g b dans la version calculatoire.

Problèmes sous-jacents à la cryptographie sur les réseaux euclidiens :

The Shortest Vector Problem :

L'un des problèmes les plus fondamentaux de ce domaine est SVP (Shortest Vector Problem) qui consiste à trouver un vecteur de norme minimale pour un réseau. Le problème SVPγ (Approximate Shortest Vector Problem) consiste à trouver un vecteur au plus γ fois plus grand que le plus petit vecteur. Le facteur multiplicatif γ est appelé le facteur d'approximation et peut dépendre de la dimension du réseau. Enfin, il est important également d'introduire la version ‖décisionnelle de SVPγ, le problème GapSVPγ (Decisional Approximate Shortest Vector Problem), et SIVPγ, équivalent de SVPγ ou` il est demandé de trouver une base approximant la plus petite base à un facteur γ près. Nous concluons cette section par les définitions formelles de ces problèmes.

Définition 1 (Shortest Vector Problem SVP). Soit L un réseau euclidien. Trouver $z \in L$ tel que $\| z \| = \lambda_1(L)$.

Définition 2 (Approximate Shortest Vector Problem SVPγ). Soit L un réseau euclidien. Trouver $z \in L$ tel que $\| z \| \neq 0$ et $\| z \| \leq \gamma\lambda_1(L)$.

Définition 3 (Decisional Approximate Shortest Vector Problem GapSVPγ). Soit L un réseau euclidien avec $\lambda_1(L) \leq 1$ ou $\lambda_1(L) > \gamma$. Déterminer quel est le cas.

Définition 4 (Approximate Shortest Independent Vector Problem SIVPγ). Soit L un réseau euclidien, et B_{min} une base de ce réseau telle que $\| B_{min} \| \leq \| B' \|$ pour toute autre base B' de L, en notant $\| B \|$ le maximum des normes des vecteurs composant une base.

Trouver une base B' telle que $\| B' \| \leq \gamma \| B_{min} \|$.

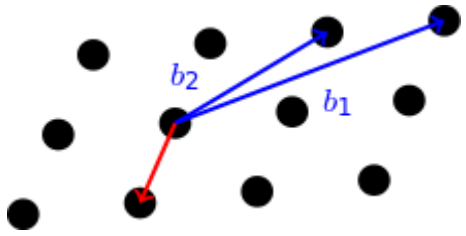


Figure 6 : En bleu la base donnée du réseau, en rouge le vecteur le plus court

The Closest Vector Problem :

Dans le domaine de la cryptographie homomorphe, on utilise fréquemment le CVP afin de créer des primitives cryptographiques sécurisées. À titre d'exemple, certains schémas de chiffrement homomorphes sont basés sur des opérations arithmétiques effectuées sur des vecteurs encodés sous forme de points dans un réseau euclidien. L'hypothèse de la sécurité de ces schémas est que la résolution du CVP dans ce réseau est complexe, ce qui rend les attaques contre ces schémas aussi complexes.

Le CVP représente un défi majeur lorsqu'il s'agit de concevoir des schémas de chiffrement homomorphes sécurisés exploitant des réseaux euclidiens. L'apparence de sa difficulté aide à

garantir la sécurité des opérations effectuées sur des données chiffrées dans le contexte de la cryptographie homomorphe.

Définition : Closest Vector Problem (CVP) : étant donnés un réseau L, une base B de L et un vecteur v, trouver le vecteur le plus proche de v appartenant à L.

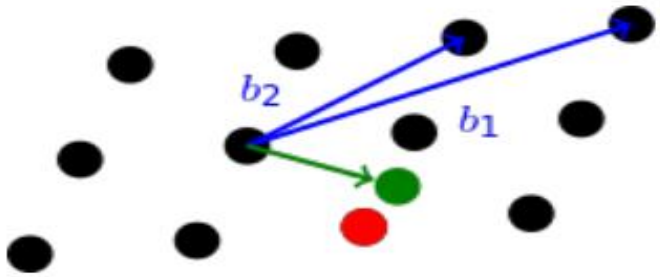


Figure 7 : En bleu la base donnée du réseau, en vert le vecteur donné, en rouge le vecteur le plus proche du vecteur vert appartenant au réseau .

Le problème LWE : Learning With Errors :

Le problème LWE (Learning With Errors) a été introduit par Oded Regev en 2005. Ce problème consiste à retrouver un vecteur secret s $\in z_n^q$ à partir d'une séquence d'équations linéaires aléatoires et "approximées" sur s. Exemple, ladite séquence pourrait être :

$$14s1+15s2+5s3+2s4\approx 8 \pmod{17}$$

$$13s1+14s2+14s3+6s4\approx 16 \pmod{17}$$

$$6s1+10s2+13s3+1s4\approx 3 \pmod{17}$$

$$10s1+4s2+12s3+16s4\approx 12 \pmod{17}$$

$$9s1+5s2+9s3+6s4\approx 9 \pmod{17}$$

$$3s1+6s2+4s3+5s4\approx 16 \pmod{17}$$

$$\vdots$$

$$6s1+7s2+16s3+2s4\approx 3 \pmod{17}$$

Dans cet exemple, s ∈ z_{17}^4 . Le ≈ signifie que chaque équation est vraie à une erreur près. C'est-à-dire qu'à chaque équation, on pourrait ajouter un terme compris entre erreur et −erreur pour que les ≈ deviennent des égalités.

Si cette erreur est égale à 0, alors les ≈ peuvent être remplacées par des =et il est aisé de retrouver s à partir des équations, en effectuant une élimination de Gauss.

Si l'erreur est faible (±1par exemple) il est encore possible de retrouver s. Mais plus l'erreur augmente et plus le problème est difficile.

Lorsque l'erreur est maximale (lorsqu'elle vaut q) alors il est impossible de retrouver s car il est impossible de le distinguer de l'erreur. Dans la séquence de l'exemple, l'erreur était de ±1, et s valait (0,13,9,11) Reformulons le problème ainsi : Soient les paramètres n≥1 (la taille), q≥2 (le modulo) et α∈[0,1] (La difficulté). Soit un vecteur secret s ∈ Z_n^q . On a en entrée un certain nombre d'équations : $a_{11}*s_1+\ldots+a_{1n}*s_n+e_1=b_1 \pmod q$

$$a_{21}*s_1+\ldots+a_{2n}*s_n+e_2=b_2 \pmod q$$

$$\vdots$$

$$a_{k1}*s_1+\ldots+a_{kn}*s_n+e_k=b_k \pmod q$$

Où les aij et les bi sont connus, les aij sont répartis uniformément et les eisont choisis aléatoirement parmi] −αq, αq [. Retrouver s est un problème difficile pour certaines valeurs de α.

<u>Le problème RLWE : Ring Learning With Errors :</u>

L'inconvénient des schémas basés sur le problème LWE est la taille mémoire nécessaire pour le stockage des échantillons ai et leurs transmissions. Une solution consiste à ajouter une structure au réseau euclidien : les idéaux de réseaux. La description des idéaux de réseaux ne nécessite qu'un vecteur au lieu d'une matrice, d'où la réduction du coût du schéma de $O(n^2)$ à O(n). On définit le problème RLWE sur les anneaux comme suit :

Soit R = Z[X]/f (x) un anneau de polynôme, f (x) = x^d + 1 un polynôme cyclotomique de degré $d = 2^k$, χ une distribution d'erreur sur R, et R_q = R/qR un anneau quotient où q ≥ 2 est un entier. Les éléments de Rq = R/qR sont les polynômes de degré inférieur à d et premier avec f (x) à coefficients modulo q. Étant donné un secret s ∈ R_q, on construit m échantillons (a, b) = (a, (a · s + e)) où a est tiré uniformément de R_q et e de χ. Le problème RLWE consiste à trouver s à partir des échantillons. La version décisionnelle du problème consiste à distinguer ces échantillons des échantillons uniformes de Z_q^2.

2.5.5 Batching/Bootstrapping

Batching :

Dans le schéma que nous avons présenté il est possible d'encoder dans un clair un vecteur dans Z ` p , en définissant le clair comme un polynôme sur Zp (au lieu d'un scalaire) ayant ` valeurs en ` points fixés à l'avance (ce qui est réalisable par exemple par une interpolation). En chiffrant ce polynôme, les opérations homomorphes seront appliquées sur chacune des ` valeurs en parallèle. Cette fonctionnalité est le batching. Elle permet d'améliorer de manière significative les performances des opérations homomorphes. En effet, plusieurs milliers d'opérations peuvent ˆêtre opérées en parallèle. On parle de full batching lorsque les opérations s'effectuent sur n valeurs pour un polynôme de degré n − 1.

Bootsrapping :

Un schéma comme celui que nous avons d'écrit rajoutera du bruit au chiffré après une opération, ou réduira l'espace disponible pour celui-ci (lors d'une réduction de module). La profondeur calculatoire sera donc limitée à moins qu'une technique de bootstrapping ne soit utilisée. Cette technique permet de réduire le bruit sans réduire pour autant l'espace disponible au-delà de ce bruit pour faire des calculs supplémentaires. Il est donc possible de refaire augmenter le bruit par des calculs, puis de le réduire à nouveau par le bootstrapping. En faisant alternativement des calculs et des appels à cette technique il est donc possible de réaliser un nombre illimité d'opérations. L'idée derrière le bootsrapping est d'évaluer de façon homomorphe la fonction de déchiffrement à partir de chiffrés de la clé. L'approche précise est complexe et nous n'avons pas besoin de la décrire dans ce manuscrit. Il nous suffit juste de noter que cette opération est coûteuse et n'est en pratique pas utilisée dans de nombreuses applications du chiffrement homomorphe. En effet, d'un point de vue calculatoire, lorsqu'il faut réaliser des calculs complexes il est généralement plus intéressant d'augmenter la taille des paramètres (et ainsi la capacite de calcul) plutôt que de faire appel au bootstrapping

2.5.6 Les types de chiffrement homomorphe :

Systémes partiellements homomorphe :

On dira d'un cryptosystème qu'il est partiellement homomorphe lorsque son espace de fonctions évaluables est une restriction de l'espace des function calculables . Un chiffrement qui n'est pas complètement homomorphe, est forcément partiellement homomorphe, dans la mesure où il peut permettre la réalisation que d'une seule opération à la fois sur des chiffrés tel que l'addition ou la multiplication, ou bien il accepte plus d'une opération mais avec un nombre limité d'itérations (pas plus d'une seule addition ou multiplication ou pas plus d'un bit).

Un chiffrement homomorphe est multiplicatif si :

$Enc(x \otimes y) = Enc(x) \otimes Enc(y)$

$\prod_{i=1}^{n} Enc(m_i) = Enc(\prod_{i=1}^{n} m_i)$

Alors $Des\ (Encp(m) \times Encp(n) = m \times n.$

Un chiffrement homomorphe est additif si :

$Enc(x \otimes y) = Enc(x) \oplus Enc(y)$

$\prod_{i=1}^{n} Enc(m_i) = Enc(\sum_{i=1}^{n} m_i)$

Alors

$Dec_{sk}(Enc_{pk}(m) \times Enc_{pk}(n) = m + n$

Exemple de chiffrement homomorphe multiplicatif :

RSA :

Auteurs : Léonard ADLEMAN [1] - Ronald RIVEST[2] - Adi SHAMIR [3]

Année : 1978

Génération des clés : Alice génère deux grands nombres premiers p et q et calcule N = pq. Elle choisit aléatoirement e premier avec $\varphi(N) = (p-1)(q-1)$ et calcule $d = e^{-1} \bmod \varphi(N)$

Clé publique :(N,e)

Clé privée : (d,N)

Chiffrement : Bob chiffre son message $m \in Z_N = Z/NZ$ en calculant $c = m^e \bmod N$, et l'envoie à Alice.

Déchiffrement : Alice déchiffre le message reçu en calculant $c^d = m^{ed} \bmod N$

Problème sous-jacent Problème RSA (hypothèse plus forte que la factorisation) Avantages : A l'époque, c'est le premier cryptosystème à clé publique!

Inconvénients : Le chiffrement est déterministe, donc non sémantiquement sûr. Des versions probabilistes existent (OAEP/OAEP+), mais elles font perdre le caractère homomorphe du système...

Homomorphie : Si $c_1 = Enc_{pk}(m_1)$ et $c_2 = Enc_{pk}(m_2)$, alors

$Dec_{sk}((c_1 \times c_2) \bmod N) = (m_1 \times m_2) \bmod N$

El Gamal :

Auteur : Taher ELGAMAL [4]

Année : 1983

Génération des clés : Alice génère un grand nombre premier p tel que $p - 1$ ait au moins un large facteur premier. Elle choisit un générateur g de Z_p. Elle choisit également un x uniformément au hasard dans $\{0,...,p-1\}$ et calcule $h = g^x \bmod p$.

Clé publique : (g,h,p)

Clé privée : x

Chiffrement : Pour chiffrer son message $m \in Z_p$, Bob choisit uniformément au hasard

$y \in \{0,...,p-1\}$, calcule $c_1 = g^y \bmod p$ et $c_2 = mh^y \bmod p$ et envoie (c_1,c_2) à Alice.

Déchiffrement : Alice calcule k = c_1^x mod p et déchiffre son message en calculant $m = c_2K^{-1} \bmod p$

Problème sous-jacent : Problème décisionnel de Diffie-Hellmann (hypothèse plus forte que le logarithme discret)

Avantages : Permet de faire de l'échange de clé simplement

Inconvénients : Expansion du message (×2)

Homomorphie : Si $(c_{1,1},c_{1,2}) = Enc_{pk}(m_1)$ et $(c_{2,1},c_{2,2}) = Enc_{pk}(m_2)$, alors

$Dec_{sk}((c_{1,1} \times c_{2,1} \bmod p, c_{1,2} \times c_{2,2} \bmod p)) = (m_1 \times m_2) \bmod p$

Exemple de chiffrement homomorphe additif :

Paillier :

Auteur : Pascal PAILLIER [5]

Année : 1999

Génération des clés : Alice choisit deux grands nombres premiers p et q et calcule

N = pq et $\boldsymbol{\lambda} = \boldsymbol{\lambda}(N) = ppcm(p-1, q-1)$ Elle choisit g aléatoirement tel que :

$pgcd(\frac{g^{\lambda} \bmod n^2 - 1}{N}, N) = 1$

Clé publique : (g,N)

Clé privée : (p,q)

Chiffrement : Pour chiffrer son message $m \in Z_N$, Bob choisit uniformément au hasard $r \in Z_N^*$

calcule et envoie $c = g^m r^N \, mod \, N^2$ à Alice.

Déchiffrement : Alice déchiffre le message reçu en calculant m = $\frac{c^\lambda \, mod \, N^2}{g^\lambda \, mod \, N^2}$ mod N

Problème sous-jacent : Problème décisionnel de résidualité composée (hypothèse plus forte que la factorisation)

Avantages : Homomorphie

Inconvénients : Expansion du message $(\times 2)$

Homomorphie : Si $c_1 = Enc_{pk}(m_1)$ et $c_2 = Enc_{pk}(m_2)$, alors

$Dec_{sk}((c1 \times c2) \bmod N^2) = (m1 + m2) \bmod N$ et $Dec_{sk}(c_2^{m_1} \bmod N^2) = (m1 \times m2) \bmod N$

Goldwasser-Micali :

Auteurs : Shafi GOLDWASSER[6] - Silvio MICALI [7]

Année : 1983

Génération des clés : Alice génère deux grands nombres premiers p et q et calcule N = pq. Elle choisit aléatoirement un non-résidu quadratique y tel que son symbole de Jacobi soit +1, c'est-à-dire $\forall z \in ZN, z^2 \bmod N \neq y$ et $(N^y) = +1$

Clé publique : (N,y)

Clé privée : (p,q)

Chiffrement : Bob décompose son message en binaire : $m = (m_0,\ldots,m_{k-1}) \in Z_2^k$ et le chiffre en $c = (c0,\ldots,ck-1) \in Z_N^k$ de telle sorte que $\forall 0 \leq i \leq k-1, ci = x^2 z^{mi} \bmod N$, où $x \in Z_N$ est choisi uniformément au hasard.

Déchiffrement : $\forall\, 0 \leq i \leq k-1, m_i = 1$ si ci est un carré modulo N, 0 sinon.

Problème sous-jacent : Résidualité quadratique (hypothèse plus forte que la factorisation)

Avantages : Pas de fuite d'information

Inconvénients : Expansion du message (×N)

Homomorphie : Si $c_1 = Enc_{pk}(m_1)$ et $c2 = Enc_{pk}(m_2)$, alors

$Dec_{sk}((c_1 \times c_2) \bmod N) = m_1 \oplus m_2$ où $\oplus$ correspond à un OU exclusif.

Systémes totalement homomorphe :

Dans la section précédente, les crypto systèmes décrits ne permettent de réaliser qu'un seul type d'opération homomorphiquement, par exemple pour RSA il s'agit de la multiplication. Dans le cas où l'espace des fonctions évaluables est l'espace des fonctions calculables (définies par exemple sous la forme d'un circuit), on dit alors que le chiffrement est totalement homomorphe. (Fully Homomorphic Encryption ou FHE en anglais).

Le chiffrement complétement homomorphe peut réaliser tout type de calcul sur les données chiffrées stockées dans le Cloud sans les déchiffrer. L'application de ce chiffrement constitue une brique importante dans la sécurité du Cloud. Plus généralement, on pourrait sous-traiter des calculs sur des données confidentielles à des serveurs situés dans Cloud tout en gardant la clé secrète qui permet de déchiffrer le résultat du calcul. C' est un cryptosystéme permettant de faire des calculs sur les données chiffrées sans les déchiffrer Formellement, si $c1$ (respectivement$c2$) est un chiffré de $m1$ (respectivement$m2$), il existe deux opérations □ et ∘ telles que : $Dec(c1 \square c2) = Dec(c1) \circ Dec(c2) = m1 \circ m$ Typiquement, ◦ sera une addition ou une multiplication modulaire, mais ce n'est pas toujours .

Les fonctions de chiffrement completement homomorphe peuvent être exprimée comme des polynômes et qu'un polynôme consiste en une série d'additions et de multiplications, un système de chiffrement sera complètement homomorphe dès lors qu'il permettra d'évaluer un nombre arbitraire d'additions et de multiplications sur les données chiffrées

Il existe aussi des constructions dites « presque homomorphes » (somewhat homomorphic en anglais) où les opérations d'additions et de multiplications sont possibles, mais pas nécessairement pour un nombre arbitraire d'opérations. Par exemple le système de Boneh-Goh-Nissim, à base de couplage, qui permet d'évaluer au plus une multiplication (pour un nombre arbitraire d'additions) .

Exemple 1 : Craig Gentry

Pour chiffrer un message, l'idée est d'y ajouter du bruit, c'est-à-dire des petites erreurs. La clé secrète permet de supprimer ce bruit, à condition qu'il ne soit pas trop gros. Les opérations homomorphes qui sont effectuées impactent également ce bruit, les bruits vont grossir. On ne pourra déchiffrer le message que si les bruits initiaux sont choisis très petits. Pour dépasser cette limitation sur le nombre d'opérations, et lorsque le bruit devient trop important, Gentry applique

la méthode de "bootstraping" ou d'amorçage. Si le déchiffrement était suffisamment efficace, on pouvait alors changer de clé publique pour réduire le bruit. On commence par utiliser une première clé, puis, quand le bruit devient trop important, on utilise une seconde clé pour rechiffrer le même message. Le bruit ou (l'erreur) e dans un idéal I d'un anneau R est définit par : $e = kI \in I \subset R$.

Le message est alors chiffré en ajoutant ce bruit au message, $En(m) = c = m + kI$

La procédure de déchiffrement consiste à retirer l'erreur. Les propriétés homomorphes du système sont réalisées, pour :

$c_1 = m_1 + k_1 I$ et $c_2 = m_2 + k_2 I$

Donc :

$c_1 + c_2 = (m_1 + m_2) + (k_1 + k_2)$ Et $c_1 \times c_2 = (m_1 m_2) + (m_1 k_1 + m_2 k_2 + k_1 k_2)$

On peut déjà remarquer que le bruit est beaucoup plus affecté par une multiplication que par une addition. Approximativement, une addition double le bruit alors qu'une multiplication l'élève au carré. Si un trop grand nombre d'opérations est effectué, le bruit devient trop grand et la procédure de déchiffrement retourne un message erroné. Cependant, en évaluant régulièrement la procédure de déchiffrement de manière homomorphe, on peut éviter que cela arrive, et c'est exactement ce que fait le bootstrapping : Etant donné un chiffré c de m, cette procédure retourne un chiffré c' de m où le bruit k' contenu dans c' est plus petit que le bruit k contenu dans c : $\|k'\| < \|k\|$. Cependant, pour pouvoir évaluer la fonction de déchiffrement de façon homomorphe, il est nécessaire que celle-ci soit suffisamment simple, ce qui n'est pas le cas initialement. Pour faire face à ce problème, Gentry réduit la complexité du circuit de déchiffrement en publiant un ensemble de vecteurs dont la somme d'une partie d'entre eux est égale à la clé secrète.

Exemple 2: BGV *[8] :Zvika Brakerski [9] : Craig Gentry [10] : Vinod Vaikuntanathan*

On considère l'anneau $A = Z^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle). On génère L modules $q_1 < \ldots < q_L$.

Un chiffré (chiffrant $m \in \{0, 1\}$ par rapport à une clé privée $s \in A_{q1}$) est un couple de polynômes de $A_{qi} = A/qiA$:

$c_1 = m + (\mathbf{A}s + 2e).\mathbf{r}$

$c_2 = \mathbf{A}.\mathbf{r}$, où $\mathbf{r} \in A_2^N$, $A \in A_{qi}^N$ et $e \leftarrow \chi^N$

Addition : $(c_1, c_2),(c_1', c_2') \in A_{qi}^2$, $(c1 + c_1', c2 + c_2') \in A_{qi}$.

Multiplication : $(c_1, c_2),(c_1', c_2') \in A_{qi}^2$, on fait le produit tensoriel :

$(c_1, c_2) \otimes (c'_1, c'_2) = (c_1.c'_1, c_1.c'_2 + c_2.c'_1, c2.c'_2)$

Le principe générale :

Clé privée : s

Clé publique : $(As + 2e, -A)$, où $e \in \chi$

Chiffré de $m \in \{0, 1\}$: $(c_1 = m + (As + 2e)\, r, c_2 = Ar) \in A_q^2$

Déchiffrement : $c_1 - c_2 s \bmod q = m + 2er$

Addition de deux chiffrés :

$c_1 + c'_1 = m + m' + As(r + r') + 2(er + er')$

$c_2 + c'_2 = A(r + r')$

Condition nécessaire pour déchiffrer : $2(er + er') + m + m' < q$

Conclusion :

Après avoir examiné l'état de l'art du chiffrement homomorphe, nous pouvons constater une avancée significative dans les méthodes de préservation de la vie privée et de la sécurité des informations. Des difficultés mathématiques complexes comme le LWE et le RLWE ont été repérées et étudiées, ce qui ouvre la voie à des solutions novatrices pour concevoir des cryptosystèmes solides et performants. L'amélioration des performances et de la praticité des opérations homomorphes a également été favorisée par les progrès en matière de batching/bootstrapping. Enfin, la division en systèmes partiellement homomorphes et totalement homomorphes des cryptosystèmes met en évidence la variété des méthodes et des applications envisageables dans ce domaine. En associant ces éléments, le chiffrement homomorphe s'avère être un outil crucial pour préserver la vie privée et la confidentialité des données dans un contexte numérique en constante mutation.

3
Les protocoles

Introduction :

Les protocoles de protection de la vie privée, tels que le PIR et l'OT, jouent un rôle crucial dans la préservation de la confidentialité des données dans un monde numérique en évolution constante. Leur utilisation permet de concilier les besoins d'accès aux informations avec les impératifs de sécurité et de confidentialité, assurant ainsi le respect des droits des individus et la conformité aux réglementations en matière de protection des données.

Le fonctionnement du PIR repose sur plusieurs principes clés :

1. Confidentialité de la requête : L'utilisateur envoie une requête au serveur, mais cette requête est conçue de manière à ce que le serveur ne puisse pas déterminer quelle information est demandée. Cela peut être réalisé en utilisant des techniques de chiffrement ou d'anonymisation.

2. Confidentialité de la réponse : Une fois que le serveur reçoit la requête, il extrait les données demandées de la base de données et renvoie la réponse à l'utilisateur. Encore une fois, il est crucial que le serveur ne puisse pas déterminer quelles données précises ont été extraites.

3. Indistingabilité des requêtes : Deux requêtes différentes doivent être indiscernables pour le serveur, de sorte que le serveur ne puisse pas distinguer les requêtes venant de différents utilisateurs ou pour différentes données.

En ce qui concerne l'Oblivious Transfer (OT), il s'agit d'un protocole qui permet à un expéditeur d'envoyer une donnée à un destinataire de manière à ce que l'expéditeur ne sache pas quelle donnée a été envoyée et que le destinataire ne sache pas quelles autres données étaient disponibles. Cette technique est utilisée pour garantir la confidentialité des échanges de données, en particulier lorsque plusieurs données sont disponibles mais qu'une seule doit être choisie.

Le fonctionnement de l'Oblivious Transfer (OT) repose sur les principes suivants :

1. Confidentialité de l'expéditeur: L'expéditeur doit pouvoir envoyer des données au destinataire sans avoir connaissance de quelles données sont reçues par ce dernier. Cela garantit que l'expéditeur ne peut pas déterminer quelle information spécifique le destinataire obtient.

2. Confidentialité du destinataire: De même, le destinataire ne doit pas être en mesure de déterminer quelles autres données étaient disponibles pour l'expéditeur. Cela assure que le destinataire ne peut pas apprendre plus d'informations que ce qui lui est spécifiquement envoyé.

3. Indistingabilité des interactions: Les interactions entre l'expéditeur et le destinataire doivent être indiscernables pour un observateur extérieur. Cela signifie que quel que soit le choix de données effectué par l'expéditeur ou le destinataire, les interactions entre eux semblent identiques.

4. Garanties de sécurité: Les protocoles OT doivent fournir des garanties de sécurité solides, idéalement basées sur des principes de théorie de l'information, pour assurer que les échanges de données sont protégés contre les attaques et les violations de la confidentialité.

Le chiffrement homomorphe est souvent utilisé dans la mise en œuvre de protocoles PIR et OT. Cette technique permet de réaliser des opérations sur des données chiffrées sans les déchiffrer, ce qui est essentiel pour garantir la confidentialité des informations pendant le processus de recherche et d'extraction.

Ces protocoles offrent des mécanismes sophistiqués pour garantir la confidentialité des informations dans un large éventail de scénarios, allant des bases de données sensibles aux échanges de données entre utilisateurs. Ils jouent un rôle crucial dans la protection de la vie privée des individus et dans le maintien de la confiance dans les systèmes informatiques.

3.1 Private Information Retrieval : PIR

Généralement, pour qu'un utilisateur veuille extraire une donnée à partir d'une base de données il suffit d'indiquer l'index ou bien l'emplacement de cette donnée, c'est le cas d'une recherche normale, mais considérant l'une des cas suivants :

¬ Une base de données contenant des informations confidentielles telles que les mots de passes des utilisateurs d'un réseau social.

¬ Une bibliothèque numérique contenant des livres qui fournissent des informations qu'on veut les considérer comme confidentielles à cause de son contenu qui peut être par exemple donne une vue sur notre personnalité.

¬ Une base pharmaceutique contenant des informations sur les produits et leurs composants ainsi que leurs effets…etc.

Dans ce cas, même le serveur peut être vu comme un non-confiant ou bien un attaquant honnête alors que le client veut que le serveur ne connaisse aucune information sur la donnée extraite ni son index sur la base, ceci dans le début peut être vu impossible mais avec l'arrivé du concept RETRAIT D'INFORMATION PRIVE ou bien PRIVATE INFORMATION RETRIEVAL (PIR).

En linguistique en s'intéresse bien sur retrait d'information privé (en masculin) et non pas privée en féminin car c'est le retrait qui est privé et non pas l'information.

Définition :

Le retrait d'information privé est un protocole qui vise la confidentialité d'extraction d'information à partir d'un entrepôt de données situant dans un serveur à distant en basant sur les mécanismes de recherche d'information ainsi que de la sécurité notamment la cryptographie, ce protocole sert à identifier le serveur comme étant un fournisseur d'informations non-confiant pour cela la confidentialité assurée vise à occulter l'information de toute personne sauf le client même si cette personne est le serveur lui-même. Formellement, un tel protocole est dit protocole PIR si :

¬ L'accès et l'obtention de l'information doit être n'importe quand et indépendamment du contenu de l'information

¬ Deux retraits e1 et e2 doivent être indistingables au niveau des requêtes correspondantes Le schéma suivant illustre bien le fonctionnement du protocole de retrait d'information privé :

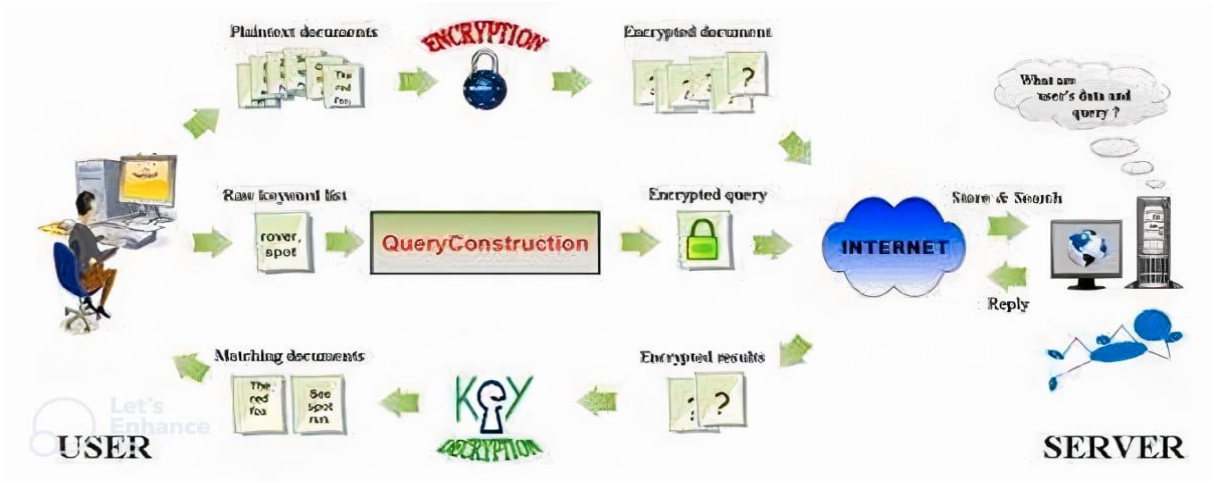


Figure 8 : fonctionnement de protocole de retrait d'information privé

principe générale :

D'après le schéma précédant on peut dire qu'un tel protocole PIR peut être réalisé en trois (03) étapes :

i. L'utilisateur génère une requête correspondante à l'information souhaitant à condition qu'elle doive être illisible au serveur
ii. Le serveur calcule la réponse qui dépend de la base de données complète et l'envoie au client
iii. Le client décode la réponse reçue et obtient l'information qu'il veut.

Il est bien sûr possible de réaliser un PIR naïf en t´téléchargeant tout le contenu de la base de données puis en choisissant localement de ne lire que l'´élément souhaite. Le premier PIR avec un coût sous-linéaire (par rapport à la taille de la base de données) a été introduit par Chor, Goldreich, Kushilevitz, and Sudan en 1995[25]. Ils proposent une manière de réaliser le PIR en utilisant des bases de données répliquées ne communiquant pas entre elles. La sécurité de ce PIR repose sur la théorie de l'information (autrement dit, il est sûr même face à un attaquant ayant une capacité de calcul illimitée). Il est possible de d´finir un protocole PIR avec un coût de communication sous-linéaire en la base de donnée et sans que celle-ci soit répliquée au prix de ne résister qu'aux attaquants ayant une capacité calculatoire limitée. On appelle ceux-ci les protocoles cPIR pour *computationally-Private Information Retrieval* [11] Une manière simple d'obtenir un tel protocole est d'utiliser un système de chiffrement homomorphe ayant la propriété d'indistingabilité face aux attaques à clair choisi.

Absorptions

Avant de d´écrire un protocole cPIR commençons par d´écrire comment absorber un clair de taille arbitraire dans un chiffrement homomorphe disposant d'une fonction d'absorption. Supposons que cette fonction prend en entrée un entier k et un chiffré c associé `a un message m et retourne Absorption (k, c), un chiffré de k · m mod p pour un entier p donné. Le comportement d'une telle fonction est remarquable pour les valeurs .

Algorithme 1 : Absorption(e,**c**,A)

Input : e élément d'une base de taille l, **c** chiffré, A capacité d'absorption d'un chiffré
Output : Vecteur de chiffrés **C** correspondant à l'absorption de e par **c**

1 Découper e en entiers de A bits e_1,...,$e_{dl/Ae}$ (en ajoutant un padding si l/A n'est pas un entier)
2 Retourner **C** = (Absorption(e_1,**c**),...,Absorption($e_{dl/Ae}$,**c**))

---

m = 1 et m = 0. En effet, si m = 1 le résultat de l'absorption sera un chiffré de k mod p (on dira que k a été retenu) alors que si m = 0 le résultat sera un chiffré de 0 (on dira que k a ´été effacé). Pour que k soit correctement retenu il faut que k < p où sinon il y aura une perte d'information à cause du modulo p. Ainsi nous d´définissons pour cette fonction une borne supérieure pour l'absorption A que par convenance nous d´définirons en nombre de bits. Enfin il est important de noter que l'on peut d´définir A < log2 p pour d'autres raisons, en particulier à cause d'une capacité de calcul homomorphe limitée (comme pour le schéma décrit dans les sections précédentes où on disposait d'un espace limité pour le bruit). Si un chiffré ne permet pas d'absorber entièrement un ´élément de la base de données (car l'´élément est trop grand en taille et provoquera une erreur lors du déchiffrement), alors le chiffré envoyé par le client est réutilisé pour absorber l'´élément par morceaux en générant ainsi une suite de chiffrés ayant absorbé peu à peu l'´élément complet. L'algorithme 1 montre comment réaliser l'absorption d'éléments plus grands que la capacité d'absorption.

Protocole simple : Soit une base de données ayant L éléments de l bits chacun. Un client crée une requête constituée de L − 1 chiffrés de 0 et un chiffré de 1. Les chiffrés sont ordonnés et le chiffré de 1 est à l'index correspond à l'´élément que le client veut récupérer. Il envoie la requête au serveur. Grâce à la propriété d'indistingabilité le serveur ne peut pas savoir à quelle position correspond le chiffré de 1. Le serveur réalise L absorptions entre chaque élément de la base de données et chaque élément de la requête. Puis il va sommer le résultat de toutes les

absorptions. Seulement l'élément de la base de données multipliant le chiffré de 1 sera retenu, les autres seront effacés, et donc la réponse contiendra uniquement l'élément souhaite. Un exemple pour une base de données à six éléments est donné dans la figure 9 et une description formelle des algorithmes associes dans la figure 10

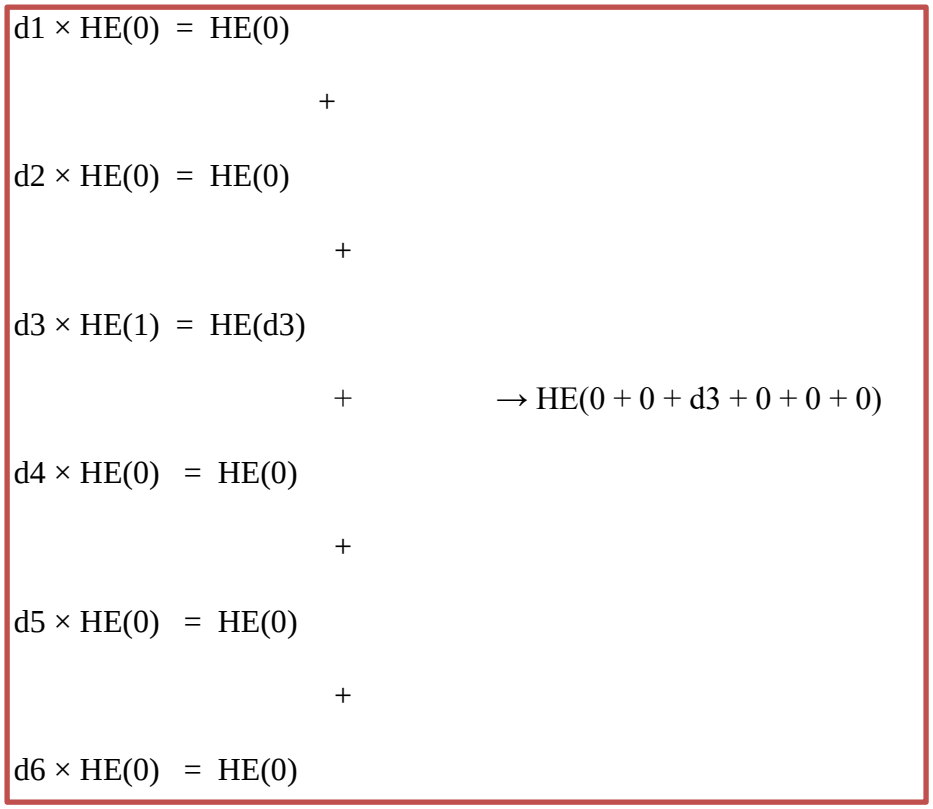


Figure 9: Génération d'une réponse PIR.

Description Fig 9 : Le client envoie un chiffré par élément dans la base de données. Le chiffré de 1 correspond à l'élément qui intéresse le client. La réponse contient un chiffré de d3 mais il n'est pas possible de savoir ceci à partir du requeté car les chiffrés de 0 et de 1 sont indistingables. Il est important de remarquer que bien que tous les chiffrés de 0 soient notés HE (0) ce sont tous des chiffrés différents appartenant à l'ensemble des chiffrés possibles de 0.

Algorithme 2 : Génération de requête cPIR

Input : Nombre d'éléments dans la base de données *L*, index i_0 de l'´élément a` obtenir, *sk* clé utilisée pour chiffrer

Output : vecteur de chiffrés Q Formant la requête

1 for i ∈ [1.....L] do
2 Qi = chiffrement (sk, 1) si i = i0 ; Qi = Chiffrement (sk, 0) sinon ;
3 Retourner Q = (Q1,.....,Q L) ;

Algorithme 3 : Génération de réponse cPIR

Input : Eléments de la base de données ($\acute{e}_1, ..., e_L$), requête $Q = (Q_1, ..., Q_L)$, capacité d'absorption des chiffrés A

Output : Vecteur de chiffrés R formant la réponse

1 Retourner $R = \sum_{i=1}^{L}$ Absorption(e_i,Q_i,A)

Algorithme 4 : Extraction de la réponse cPIR

Input : Réponse PIR R formée des chiffrés $R_1, ..., R_{\lceil dl/Ae \rceil}$, clé de d´déchiffrement sk, capacité d'absorption des chiffrés A

Output : Elément de la base de données e

1 **for** $i \in [1...\lceil dl/Ae \rceil]$ **do**

2 partie$_i$ = déchiffrement(sk,R_i);

3 Concaténer les partie$_i$ comme des éléments de A bits en ´éliminant le padding du dernier le cas échéant;

4 Retourner le résultat;

Figure 10:Algorithmes formant un PIR calculatoire fondé sur un chiffrement homomorphe.

3.2 Oblivious Transfer : OT

Un *Oblivious Transfer* (OT) est un protocole qui permet d'envoyer une donnée sans que l'expéditeur n'apprenne quelles informations ont ´été transmises. Contrairement aux protocoles PIR ces protocoles sont toujours symétriques, ils doivent garantir que l'utilisateur ne peut apprendre qu'une des données de l'expéditeur par transaction. Contrairement encore aux protocoles PIR, le cout des communications n'est pas un critère prioritaire et généralement on considère que la base entière (chiffrée) est envoyée.

Les objectifs de l'OT sont de donner des garanties le plus fortes possibles de sécurité (idéalement fondées sur les principes de la théorie de l'information) et de minimiser les couts calculatoires.

Le premier schéma d'OT a ´été proposé en 1981 par Michael O. Rabin mais la forme la plus utilisée des schémas OT correspond à un schéma ultérieur publié par Even et al. en 1985, le 1-2 OT ou *1-out-of-2 oblivious transfer*. Dans ce protocole on envoie 2 éléments chiffrés et le destinataire ne peut déchiffrer qu'un des deux.

Definition :

(*1 out of 2 Oblivious Transfer* 1-2 OT)**.** *Un schéma d'1-2 OT est donne par deux algorithmes interactifs : l'expéditeur S et le destinataire R. L'expéditeur prend en entrée deux messages* m_0 *et* m_1*, le destinataire un bit b. Le destinataire reçoit* m_b *mais n'apprend pas d'information sur* $m_{b+1 \bmod 2}$**.** *Les interactions pour* $b = 0$ *et* $b = 1$ *sont indistingables. L'expéditeur n'apprend pas d'information sur b et ne sait donc pas quel message le destinataire n'a reçu.*

L'OT a été généralise a une situation ou` le destinataire choisit parmi *n* éléments. L'utilisateur récupère toujours exactement un élément d'une base de données sans que le serveur apprenne quel élément il a récupère et sans que l'utilisateur n'apprenne les autres éléments.

Les protocoles d'OT n'utilisent pas forcement le chiffrement homomorphe et nous les utiliserons dans ce manuscrit en boite noire uniquement leur fonctionnement interne n'ayant pas d'impact sur nos protocoles.

Conclusion :

Les protocoles de protection de la vie privée sont des mécanismes et des techniques conçus pour garantir que les données personnelles et sensibles des individus sont collectées, stockées, traitées et partagées de manière sécurisée et confidentielle.

Le Private Information Retrieval (PIR) est un protocole visant à permettre l'extraction de données d'une base de données distante sans que le serveur ne puisse savoir quelle donnée est extraite ni même avoir connaissance de l'index de cette donnée dans la base. Cela garantit la confidentialité de l'extraction d'informations, même si le serveur est considéré comme non-confiant ou potentiellement malveillant.

Le PIR repose sur des mécanismes de recherche d'informations et de sécurité, notamment la cryptographie. Il permet à un client d'obtenir une information souhaitée de manière indépendante du contenu de l'information et en préservant l'indistinguabilité entre différentes requêtes.

Un exemple de mise en œuvre du PIR est donné par l'utilisation de chiffrement homomorphe, qui permet d'effectuer des calculs sur des données chiffrées sans les déchiffrer, assurant ainsi la confidentialité de la requête.

L'Oblivious Transfer (OT) est un autre protocole symétrique qui permet d'envoyer des données sans que l'expéditeur ne sache quelles informations ont été transmises, et où le destinataire ne peut obtenir qu'une seule des données envoyées, sans que l'expéditeur ne sache laquelle.

Les protocoles d'OT sont utilisés pour garantir des niveaux élevés de sécurité tout en minimisant les coûts calculatoires. Ils sont souvent utilisés dans des situations où le destinataire doit choisir parmi plusieurs éléments sans que l'expéditeur ne puisse connaître le choix effectué.

Le PIR et l'OT sont des protocoles de sécurité qui permettent respectivement l'extraction confidentielle d'informations à partir d'une base de données distante et l'envoi de données sans que l'expéditeur ne connaisse les informations transmises ni le destinataire ne connaisse les autres informations disponibles.

4

Protection de la vie privée :Applications

Introduction :

Dans un paysage numérique en constante évolution, la préservation de la confidentialité et de la sécurité des données sensibles est devenue une préoccupation majeure pour les entreprises et les institutions du monde entier. Deux domaines en émergence, bien que distincts, posent des défis similaires en matière de gestion et de protection des données : le stockage sécurisé des données génomiques et de séquençage ADN, ainsi que la détection de fraudes dans les transactions en ligne.

Dans cette introduction, nous plongerons dans les enjeux cruciaux associés à ces deux domaines et proposerons des solutions innovantes pour répondre à ces défis tout en préservant la vie privée des individus.

La première partie de ce chapitre se concentrera sur le stockage sécurisé des données génomiques et de séquençage ADN. Alors que les avancées technologiques dans le domaine de la génomique ont conduit à une explosion de données sensibles, la nécessité de les protéger tout en permettant leur utilisation à des fins de recherche et d'analyse est devenue primordiale. Nous présenterons une approche novatrice permettant de vérifier la présence de données dans une base de données chiffrée, tout en préservant leur confidentialité, offrant ainsi une solution sécurisée et efficace pour la gestion des données génomiques.

Dans la seconde partie, nous aborderons les défis liés à la détection de fraudes dans le commerce électronique. Avec l'essor du commerce en ligne, la détection et la prévention des activités frauduleuses sont devenues des impératifs commerciaux. Nous explorerons un algorithme de détection de fraude privée qui sécurise les données sensibles tout en permettant une détection précise des transactions suspectes, assurant ainsi la confiance des clients tout en protégeant leur vie privée.

En combinant des approches techniques avancées avec un engagement résolu envers la protection de la vie privée, ce chapitre offre un aperçu des solutions prometteuses pour relever les défis complexes posés par la gestion des données sensibles dans un monde numérique en constante évolution.

4.1 Sécurité de données génomiques :

Représentation des données génomiques Un chromosome est représenté comme une suite d'allèles : adénine (A), cytosine (C), guanine (G), et thymine (T). Un génome représente l'ensemble des chromosomes. Il existe un génome de référence par rapport auquel il est possible de décrire un autre génome, en n'indiquant que les différences. L'intérêt de cela est qu'encoder l'ensemble des différences avec un génome est beaucoup moins lourd que d'encoder un génome tout entier car les génomes de deux humains quelconques sont identiques pour l'immense majorité des allèles. [12] En effet, comme déjà dit, pour stocker un génome explicitement il faut environ 100Gb. La taille n´nécessaire pour une représentation ne décrivant que les différences entre le génome d'un individu et un génome de référence dépendra du nombre de mutations que possède l'individu mais ne dépassera pas les quelques Mb. Il suffit d'indiquer le chromosome, la position au sein du chromosome et la nature de la différence : cette différence est appelée mutation. Il existe 4 types de mutations différentes : la substitution, la délétion, l'insertion et la substitution multiple.

Une substitution correspond à une différence sur un allèle. Le client possède `a une position donnée un allèle différent par rapport au génome de référence. Par exemple GATACA peut devenir GATTCA.

La délétion correspond à une coupure d'un ou plusieurs allèles à partir d'une position. Par exemple GATACA peut devenir GATA.

L'insertion correspond `à un ajout d'un ou plusieurs allèles à partir d'une position. Par exemple GATACA peut devenir GATAGCCA.

La substitution multiple est une différence sur plusieurs positions consécutives. Par exemple GATACA peut devenir GATTGA. Le format VCF Variant Call Format est un format de fichier qui permet d'encoder les mutations d'un individu. Chaque ligne du fichier correspond `à une mutation présentée sous un format précis : chromosome, position, identification, allèle de référence, type de mutation et allèles de l'individu. Ces informations permettent de d´définir la mutation quelle qu'elle soit

Le problème

Nous avons vu qu'il ´était important de se soucier de la confidentialité des données génomiques, ainsi que du contrôle de leur accès. Le contexte dans lequel nous nous plaçons est le suivant :

un docteur possède les données issues du s´séquençage du génome d'un patient, sous la forme d'un fichier VCF, et veut le stocker en ligne sans que personne, y compris le serveur, ne puisse y accéder. Il est également nécessaire de cacher la façon dont on accède à ces données. En effet, en fonction de qui accède `a une donnée et la fréquence d'accès, il est possible d'apprendre des informations. Par exemple un accès aux mutations ayant une influence sur l'efficacité de certains traitements pour le cancer de prostate peut donner des indications sur la maladie (cancer) sexe (masculin) et âge (probablement supérieur à 65 ans) du patient correspondant à un génome. Afin de pouvoir garantir un niveau élevé de sécurité, nous avons décidé d'avoir la possibilité de cacher le nombre de mutations que le client possède. On peut supposer qu'un nombre de mutations anormalement ´élevé pourrait entrainer des préjudices au client. Il est donc n´nécessaire de cacher cette information. Par ailleurs, il est n´nécessaire de cacher la taille des mutations afin de ne pas pouvoir faire de corrélation entre une mutation et sa taille. Le m´médecin doit pouvoir réaliser une requête et recevoir une réponse sans échange intermédiaire. La requête identifie une mutation et la réponse est un booléen indiquant si le client possède ou non la mutation identifiée. Pour récapituler :

Le serveur ne doit pas avoir accès aux données en clair ;

Le serveur ne doit pas apprendre quelles informations sont accédées ;

Le serveur ne doit pas apprendre quelle est la taille des données (nombre et taille des mutations)

Une requête doit permettre d'apprendre si une mutation est présente ou pas ;

La réponse est un booléen qui indique la présence de la mutation.

La solution

Notre solution est fondée sur un systèmes de chiffrement homomorphe et un PIR. Le client a pour entrée un fichier VCF. Une ligne correspond à une mutation. Cependant l'encodage d'une mutation peut être particulièrement grand. En effet, il existe des insertions de plusieurs centaines d'allèles. Comme déjà expliqué, nous ne pouvons pas insérer des données de tailles différentes dans la base de données car cette information peut permettre d'identifier partiellement ou totalement un client. C'est pour cette raison que le client applique une fonction de hachage sur chaque mutation. Pour réduire la taille de la base de données le client ne va conserver qu'une partie de l'haché obtenu. Notons h = h1...hy, avec y la taille de l'haché. Les bits h1...hx serviront d'index dans la base de données. Ensuite, hx+1...hy est ajouté dans la base

à l'index indiqué. S'il y a déjà des données stockées `a cet index, le nouvel ´élément est ajouté au même index, de telle façon qu'à chaque index sera associée une liste d'éléments (chacun de ces éléments étant un hachage tronqué d'une mutation). Une fois toutes les mutations insérées dans la base de données locale, le client va réaliser un bourrage pour simuler la présence de 5 millions de mutations dans la base de données (aucun ˆêtre humain n'a autant de mutations connues pour le moment) [61], cela permet d'obtenir le même nombre de mutations pour chaque client. Puis il va réaliser l'empreinte cryptographique de ce bourrage et l'insérer avec le même processus. Pour que chaque ligne contienne le même nombre d'éléments, le client va ensuite réaliser un autre bourrage en ajoutant des éléments aléatoires jusqu'à ce que chaque index contienne une liste de taille identique. Il va chiffrer symétriquement, par bloc avec un mode compteur, ligne par ligne le fichier la base de données obtenue. Le client conserve en mémoire le compteur utilisé pour chaque position du chiffrement. Ce processus est décrit dans la figure 3.2

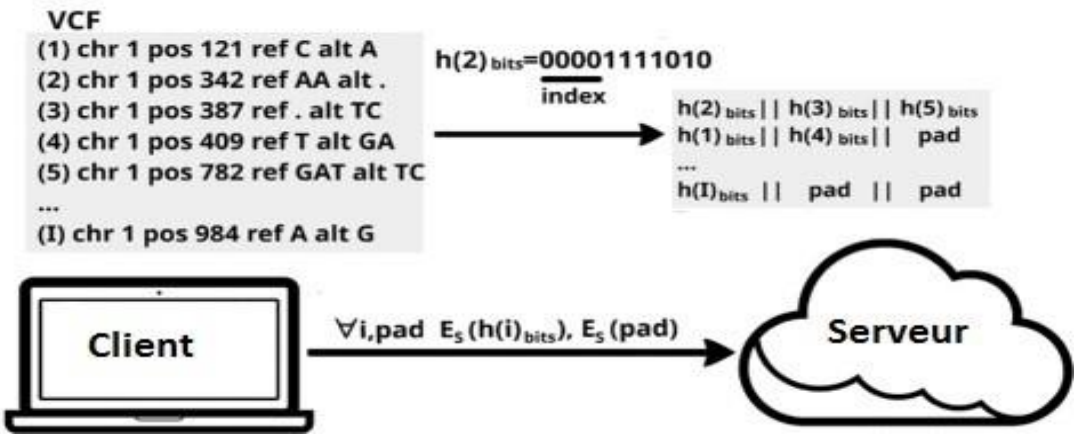


Figure 11:VCF Client/Serveur

Description Fig 11 : Lors de la phase d'initialisation, le client va hacher des informations de chaque ligne du fichier VCF. A chaque ligne sera associée une empreinte. Il va utiliser les premiers bits de cette empreinte pour d´définir l'index de la base de données ou` il enregistrera cette empreinte. Il va finalement chiffrer symétriquement chaque empreinte obtenue avant de l'enregistrer à son index. Il fera du bourrage pour simuler un fichier VCF de taille 5 millions de mutations puis fera du bourrage pour que chaque entrée de la base de données fasse la même taille. Pour simplifier, cette image ne décrit pas comment les hachés sont tronqués pour réduire la taille de la base de données.

Peut permettre d'identifier partiellement ou totalement un client. C'est pour cette raison que le client applique une fonction de hachage sur chaque mutation.

Le client envoie ensuite cette base de données chiffrée qui sera stockée par le serveur. Pour savoir si une mutation est présentée ou non, le client va générer une requête PIR. Pour cela, il va reproduire le processus de la phase d'initialisation pour obtenir l'empreinte cryptographique de la mutation. Grâce à cela le client détermine la ligne qui peut contenir l'information concernant la mutation qu'il recherche. Le client va envoyer sa requête PIR pour télécharger toute la ligne qui contient l'information sur la mutation.

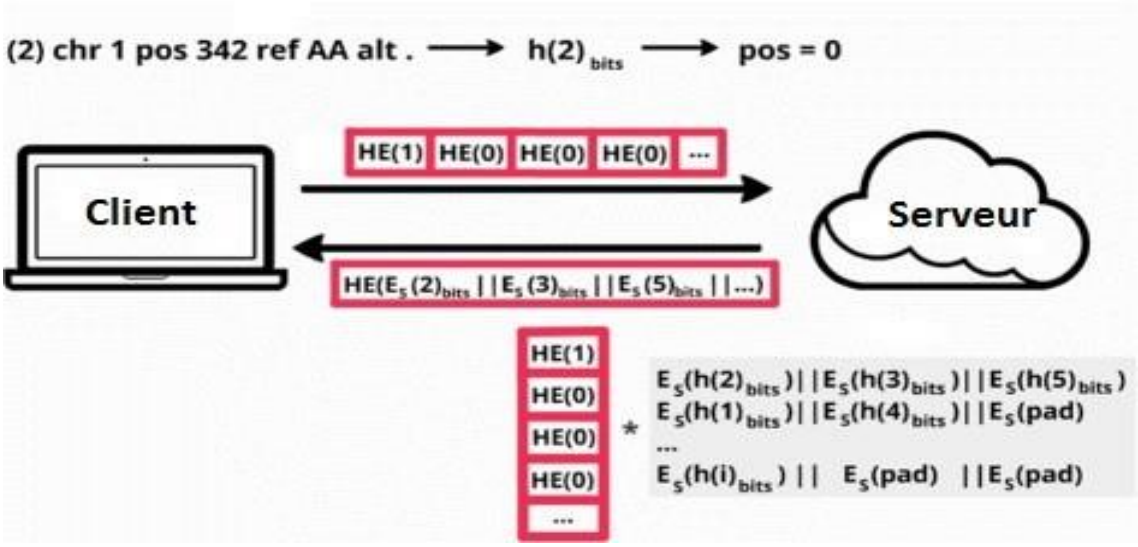


Figure 12:Demande de la mutation

Description Fig 12: Lors de la phase de requête, le client forme une requête PIR pour savoir si une mutation est présentée ou pas. Pour cela il va hacher la mutation qui l'intéresse puis réaliser une requête PIR pour récupérer la ligne ayant pour index les premiers bits de l'empreinte obtenue.

Cependant, si l'on réalise un simple PIR, le client va récupérer beaucoup trop d'informations, ce que l'on ne veut pas n´nécessairement. En effet, puisque les données sont agrégées dans la base de données, si le client récupère une ligne entière, il va apprendre tous les éléments contenus sur cette ligne. Si on suppose par exemple que la personne qui fait la requête est un médecin et pas le patient lui-même, il est préférable de cacher le plus d'informations possible. Pour cela, en plus de la requête PIR, le client va également envoyer un chiffré, qu'on appellera requête de soustraction, qui va permettre d'obfusquer la réponse du serveur. Afin de comprendre cette obfuscation, il est nécessaire de donner quelques précisions sur le protocole PIR utilisé. Nous avons choisi d'utiliser un protocole PIR fondé sur RLWE en full batching. Comme vu précédemment avec RLWE les clairs sont des

polynômes (qu'il est possible de voir comme des vecteurs d'évaluations en différents points de ces polynômes). Dans notre protocole, les réponses PIR sont donc des vecteurs chiffrés, et nous paramétrons le protocole de telle façon qu'à chaque coordonnée de ces vecteurs est associé un unique haché (tronqué et chiffré) d'une mutation. La requête de soustraction est un chiffré RLWE en full batching qui contient l'empreinte chiffrée de la mutation recherchée à chaque coordonnée. A la réception de ce chiffré le serveur fait une soustraction entre la réponse PIR et la requête de soustraction. Si la mutation est présentée dans la réponse, cette opération va faire apparaitre un zéro . Pour masquer les autres informations le serveur randomise chaque coordonnée . Si l'une de ces coordonnées vaut zéro elle est inchangée par cette opération, mais toutes les positions qui ne sont pas à zéro (c'est-`a-dire ne correspondant pas exactement à la mutation recherchée) seront obfusquées par un masque multiplicatif inconnu du client. Une fois le résultat reçu, le client va le déchiffrer et chercher si la réponse possède un 0 à n'importe quelle position. Si c'est le cas, la mutation est présentée. Le client apprend également la position à laquelle la mutation a été stockée. Pour s'assurer que cela ne révèle pas d'informations il est possible de modifier ce protocole de façon à ce que le serveur change aléatoirement l'ordre des éléments au sein de chaque index .

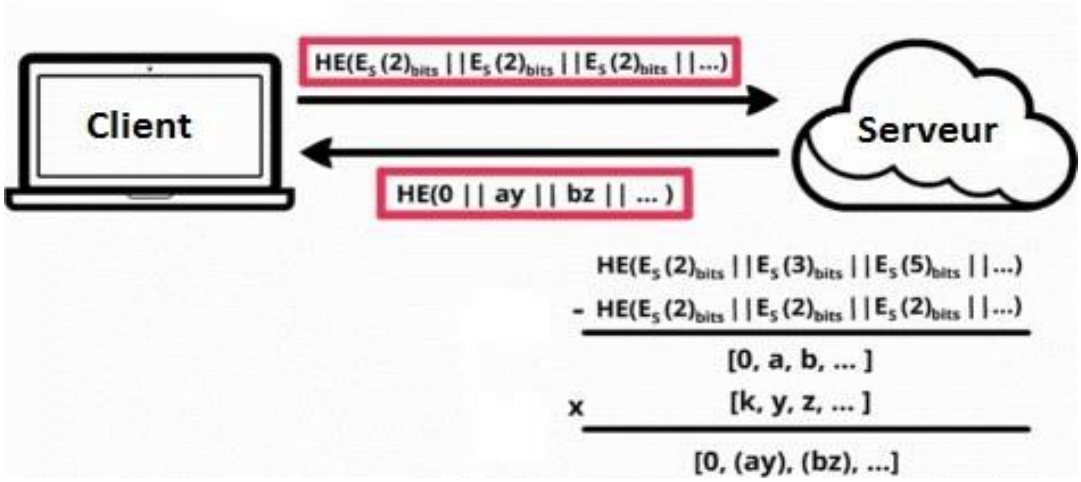


Figure 13:Elément de mutation

Description Fig 13 : *Un client réalisant une requête PIR classique sur la base de données va récupérer de nombreuses informations puisqu'une entrée dans la base de données correspond à de nombreuses mutations concaténées. Pour cacher ce surplus d'informations il va envoyer une requête de soustraction afin de faire*

apparaître un 0 dans le résultat du PIR si la mutation est présentée. Il va ensuite faire en sorte que les résultats soient multipliés par des nombres aléatoires ce qui permettra d'éliminer toute autre information que si un résultat était nul ou pas.

4.2 Détection du Fraude lors d'un paiement en ligne :

Dans cette partie, nous présentons un algorithme de détection de fraude privé qui protège les données liées à la détection de la fraude mais également les données du client.

Beaucoup d'entreprises se sont tournées vers le commerce électronique afin d'améliorer leur productivité et leur visibilité dans le but de vendre leurs produits ou services. Ces systèmes de paiement électronique sont utilisés par des utilisateurs légitimes mais également par certains autres frauduleux. La fraude est un crime dont le but est de s'approprier de l'argent par des moyens non légaux. *Inter-net Crime Complaint Centre* (IC3) est une agence qui regroupe le FBI (*Federal Bureau of Investigation*), le NW3C (*National White Collar Crime Center*) et le BJA (*Bureau of Justice Assistance*) et qui fournit aux plaignants de cybercrimes un mécanisme de déclaration pratique et simple pour alerter les autorités.

une perte de 3,5 milliards en dollars en 2019.indique L'importance des pertes dues aux fraudes qui a entrainé la recherche de techniques pour la combattre. Le principe des mécanismes de prévention de fraude est de protéger les systèmes en empêchant la fraude de se produire. Cependant, ces mécanismes ne sont pas toujours très performants et ne permettent pas seuls de prévenir au mieux la fraude. Les systèmes de détection de fraude permettent d'améliorer la sécurité contre la fraude et notamment de rapporter la fraude à l'administrateur du système. Les fraudes relatives aux cartes bleues peuvent être classées en deux catégories :

La fraude hors ligne, qui consiste à voler matériellement la carte bleue de la victime pour l'utiliser. Ce type de fraude est peu commun puisque, dans la majeure partie

Des cas, lorsque la banque a été mise au courant du vol, toutes les transactions provenant de la carte sont bloquées.

Le fraude en ligne, qui consiste à récupérer les informations liées à la carte bleue de la victime et les utiliser. Ce type de fraude est plus commun. Les informations peuvent être récupérées via une faille lors d'un paiement électronique, une campagne de phishing ou encore une génération aléatoire de carte.

La plupart des techniques pour détecter ces fraudes se basent sur la détection d'anomalies. Le profil du client est dressé en fonction de son comportement. De cette façon, toute transaction qui serait incohérente avec le profil dressé sera considérée comme suspicieuse. Les systèmes de détection de fraude utilisent en général un arbre de décision ou une décision basée sur des règles.

La prise de conscience dans le domaine de la protection de vie privée est très récente, et l'exploration de pistes pour réaliser une détection de fraude de manière privée est presque inexistante. En 2018 Canillas et al 77 . utilisent un arbre de décision qui conserve le caractère privé des données. L'approche est très intéressante mais couteuse puisqu'elle nécessite environ 1*s* pour réaliser la détection de la fraude ainsi que 2.5Go de bande passante sur le réseau .

Présentation du problème :

Détection privée

Dans la solution présentée, nous supposons que le serveur qui s'occupe de la détection de fraude possède un ensemble de règles concernant les données du client de la forme suivante : si une donnée du client x est supérieure à un seuil t alors le score de fraude du client est incrémente de α. Si le score de fraude est supérieur à un seuil final T, une alerte est levée par le serveur car le paiement est suspicieux. Nous ne nous intéressons pas à la façon dont ont été génères les seuils t et T ni les pondérations α. Ces données sont généralement obtenues en dressant le profil du client puis par l'utilisation d'algorithmes bases sur les réseaux de neurones. Le nombre de caractéristiques du client est note N. Ces données peuvent représenter par exemple le montant ou le lieu du paiement, l'historique du client, etc. La détection de la fraude s'effectue lors du paiement, nous supposons donc que les connexions nécessaires sont déjà établies pour permettre le paiement et donc la détection de la fraude. La durée nécessaire pour réaliser un paiement ne dépasse pas quelques secondes, nous souhaitons donc proposer une solution ne dépassant pas quelques centaines de millisecondes. Il est préférable, pour respecter cette contrainte temporelle, d'avoir une solution peu interactive.

Etat de l'art

Client Serveur

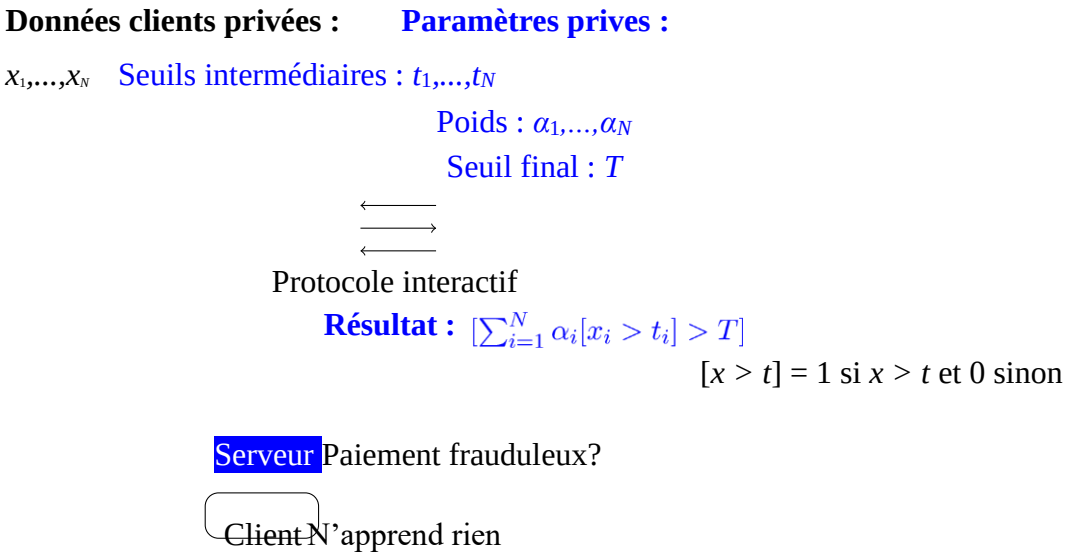


Figure 14:La sécurité des paramétres

Figure 5.1 Description des paramètres et de leur sécurité. Le client possède ses données de paiement. Le serveur possède des seuils intermédiaires qui correspondent à chaque donnée, des poids associes a` ces seuils et un seuil final. Il souhaite vérifier sans apprendre les données du client si le paiement est frauduleux, pour cela il doit vérifier pour chaque donnée de paiement si elle est plus grande que le seuil intermédiaire, puis faire la somme pondérée par les poids de ces tests d'inégalité, et enfin vérifier si cette somme est plus grande que le seuil final. Le client ne doit pas apprendre les données du serveur.

Le client pourrait être un terminal de paiement ou un service de paiement en ligne. La fraude la plus courante étant celle en ligne, nous baserons les résultats sur les capacités d'un ordinateur. Lors de la détection de fraude privée, le client ne doit apprendre ni les seuils intermédiaires t, ni les pondérations α ni le seuil final T.

Le serveur ne doit apprendre ni les données x du client, ni le résultat final du score de fraude du client, ni les résultats intermédiaires $[x > t]$. $[x > t]$ vaut 1 si x est plus grand que t et 0 sinon.

La solution pour permettre le paiement privé tel que d´écrit ci-dessus passera par des comparaisons privées. A priori, le moyen le plus rapide [78] pour réaliser les comparaisons privées est l'utilisation des *garbled circuits* [79].

La génération du circuit se fait conventionnellement comme un ensemble de portes logiques XOR et ET a` deux entrées en minimisant le nombre de ET puisqu'il existe une optimisation sur les portes XOR .

Prenons l'exemple d'une porte ET `a deux entrées pour illustrer le fonctionnement d'un *garbled circuit*. Soit la table de vérité de la porte ET ayant pour entrer *a* et *b* et comme sortie *c*.

a	*b*	*c*
0	0	0
0	1	0
1	0	0
1	1	1

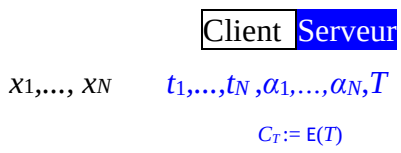
Soit Alice qui réalise le *garbled circuit*. Alice attribue des labels qui sont generes comme des nonce pour chaque élément de la table de vérité. La taille du label représente le paramètre de sécurité. Un label est génère pour chaque valeur binaire. Voici la table de vérité des labels.

a	*b*	*c*
X_{0a}	X_{0b}	X_{0c}
X_{0a}	X_{1b}	X_{0c}
X_{1a}	X_{0b}	X_{0c}
X_{1a}	X_{1b}	X_{1c}

Puis Alice chiffre cette table pour obtenir la *garbled table*. La fonction de chiffrement est une fonction symétrique qui utilise 2 clés. Les clés sont les labels correspondants aux entrées, le label correspondant a` la sortie est chiffré. Les entrées de la table subissent une permutation aléatoire. Cette table est envoyée à Bob.

Alice envoie à Bob le label correspondant a` son entrée X_0^a ou X_1^a. Bob, pour obtenir son label, va réaliser un *1 out of 2 oblivious transfer* . Bob peut ensuite déchiffrer une entrée dans la *garbled table* et envoyer le label correspondant Alice qui peut retrouver la valeur booléenne.

Une inégalité privée avec des entrées sur 8 bits s'exécute en 40ms en utilisant un *garbled circuit*. Cependant dans le protocole de paiement privé, les résultats intermédiaires doivent ˆêtre privés. Cette contrainte augmente la taille du circuit. En gardant le temps d'exécution dans l'ordre de la centaine de millisecondes, il ne serait possible que de réaliser 4 `à 5 inégalités en utilisant un *garbled circuit*.



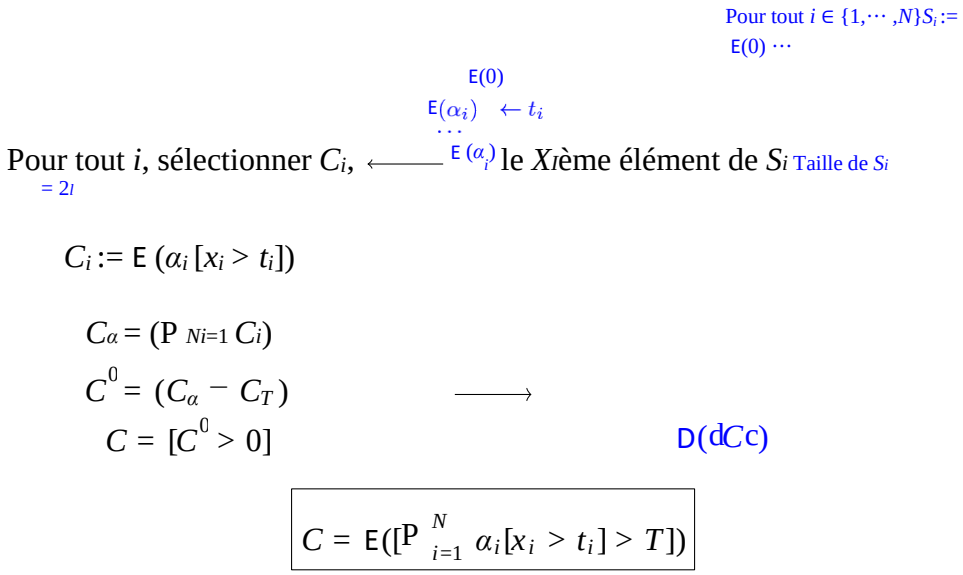


Figure 15:Protocole permettant de réaliser la détection de fraude de manière privée

Le serveur envoie, pour chaque donnée x du client de taille l, 2^l chiffres homomorphes. Ces chiffres sont des chiffres de 0 pour les t premiers puis des chiffres de α pour les suivants. Le client sélectionne le $X^{ème`}$ chiffré qui correspond au test $[x > t]$ multiplié par α. Il somme l'ensemble des résultats obtenus, soustrait le seuil final envoyé par le serveur sous forme de chiffrée homomorphe et teste si le résultat est positif ou négatif. Cette étape sera d´écrite dans l'algorithme 13. Le client envoie le résultat du test final qui est déchiffre par le serveur pour savoir s'il y a fraude.

Notre solution :
KODA

Notre solution est d´écrite dans la figure Soit l la taille d'une donnée x_i possédée par le client. Le serveur va utiliser une fonction de chiffrement homomorphe pour obtenir 2^l chiffrés qui représentent toutes les valeurs possibles de x_i. Le serveur génère t_i, le seuil intermédiaire, chiffres de 0 puis $2^l - t_i$ chiffres de α_i, le poids associe. Cet ensemble de chiffres est noté S_i, les premiers chiffres de cet ensemble sont ceux de 0. Cette opération est répétée pour les N données du client. Le serveur chiffre également, avec une fonction de chiffrement homomorphe T, le seuil final. Ces chiffres sont envoyés au client. Pour chaque ensemble le client va sélectionner le $X^{ème}$ chiffré noté C_i. Le chiffré représente $\mathsf{E}(\alpha_i [x_i > t_i])$ ou $[x_i > t_i]$ vaut 1 si x_i est plus grand que t_i et 0 sinon. En effet, le serveur a généré t_i chiffrés de 0, ainsi en sélectionnant le $X^{ème}$ chiffré, le client récupère le résultat de l'inégalité puisque si x_i est inférieur a` t_i il s´sélectionne un chiffré de 0 et sinon un chiffre de α_i.

Le client somme tous les C_i, $C_\alpha = (\sum_{i=1}^{N} C_i)$ représente le score de fraude du client.La dernière étape de l'algorithme consiste à calculer $[C_\alpha > C_T\ ]$ puis a` renvoyer ce résultat au

serveur. Si le score de fraude du client est supérieur au seuil final alors le serveur levera une fraude.

Conclusion :

Notre solution offre en effet une série d'avantages significatifs en matière de sécurité des données génomiques et de détection des fraudes, tout en veillant à respecter la confidentialité des utilisateurs :

Pour les données génomiques :

Confidentialité renforcée de la vie privée: Grâce à des techniques telles que le bourrage de fichiers et l'utilisation de hachages, les données génomiques sont protégées de manière robuste. Même en présence d'un grand nombre de mutations, leur confidentialité demeure intacte. De plus, l'utilisation du PIR dissimule efficacement l'accès aux données, garantissant ainsi leur manipulation confidentielle et préservant la vie privée des utilisateurs.

-Intégrité et confidentialité des données garanties : Chaque mutation est sécurisée par une empreinte numérique, ce qui garantit non seulement l'intégrité des données, mais aussi leur confidentialité. Ainsi, les informations génomiques restent protégées contre toute altération ou accès non autorisé.

Pour la détection de fraudes :

Protection des données confidentielles : nos données restent exclusivement accessibles à notre algorithme de détection, préservant ainsi leur confidentialité et celle des utilisateurs.

Pour améliorer encore la fiabilité et l'efficacité de notre solution, des améliorations peuvent être apportées aux protocoles et aux algorithmes utilisés. Par exemple :

Optimisation des protocoles de sécurité: En affinant et en optimisant les protocoles de sécurité, nous pouvons renforcer la protection des données génomiques contre les menaces potentielles.

Raffinement des algorithmes de détection : En continuant à développer et à perfectionner les algorithmes de détection des fraudes, nous pouvons améliorer la précision et la sensibilité de notre système, ce qui permet une détection plus rapide et plus fiable des activités suspectes.

Conclusion et perspectives

Cette recherche consiste à appliquer le chiffrement homomorphe à la protection de la vie privée, en premier lieu nous avons étudié et déterminé les outils mathématiques et les problèmes de chiffrement homomorphe de plus ses type qui peuvent nous aides à développer des solutions efficaces pour assurer la protection de la vie privée .

Après nous avons parlé des protocole suivants : PIR (Private Information Retrieval) ainsi ses différents types , et OT (Oblivious Transfer) ,ces deux derniers sont basés et améliorés par le chiffrement homomorphe

Enfin, Les données génomiques sont liées aux individus et soulèvent des problèmes de vie privée qui sont difficiles à contourner. La protection de ces données n'est pas une tâche aisée, elle est cependant capitale. Nous avons présenté une solution à un des potentiels problèmes du stockage externalisé de données génomiques. Notre solution permet, en outre du stockage confidentiel, d'interroger le serveur sur la présence ou l'absence d'une mutation dans le génome stocké. Nous assurons la confidentialité des données stockées, mais également celle de la requête, vis-à-vis du serveur. La solution présentée est très efficace d'un point de vue mémoriel et temporel. La dernière contribution du manuscrit est une solution permettant de réaliser une détection de fraude de manière privée. Ainsi, les données du client ne sont pas révélées au serveur qui s'occupe de la détection. Nous avons montré qu'il était possible de réaliser la détection sur un nombre de données suffisant en moins de 200 ms en se basant sur un protocole proche de l'OT.

Bien évidemment, nous avons rencontré plusieurs difficultés durant la réalisation de notre travail :

Essentiellement lors de la recherche de quelles notions, nous étions obligés de comprendre d’autres notions plus complexes et nécessitent des connaissances scientifiques plus avancée.

Malheureusement qu’on n’a pas arrivé à accomplir la tâche d’implémentation, car elle nécessite des machines plus performantes et couteuses.

De toute façon, ce projet était très bénéfique au niveau du plan professionnel et technologique, et c’était l’occasion pour améliorer nos connaissances et acquérir de nouveaux concepts dans le domaine de la sécurité informatique.

De point de vue perspectif, ce travail nécessite encore des amélioration à savoir :

• Amélioration des protocoles PIR par la Récursion et Agrégation

• implémentation des solutions

• Suivi en temps réel de l’activité de chaque solution

Bibliographie

[1] : **Leonard Max Adleman**, né le 31 décembre 1945 (78 ans), est un chercheur américain en informatique théorique et professeur en informatique et en biologie moléculaire à l'université de la Californie du Sud. Co-inventeur du cryptosystème RSA (Rivest, Shamir, Adleman) en 1977, Adleman a également travaillé dans la bio-informatique.

[2] : **Ronald Linn Rivest** (né le 6 mai 1947 à Schenectady1 dans l'État de New York) est un cryptologue américain d'origine canadienne-française. Il est l'un des trois inventeurs de l'algorithme de cryptographie à clé publique RSA, premier exemple de cette famille.

[3] : **Adi Shamir** (en hébreu עדי שמיר), né le 6 juillet 1952 à Tel Aviv, est un mathématicien et un cryptologue israélien reconnu comme l'un des experts les plus éminents en cryptanalyse1. Il est professeur au département de mathématiques appliquées de l'Institut Weizmann depuis 1984 où il occupe la chaire Borman de science informatique. En 1978, il a créé avec Ron Rivest et Len Adleman, l'algorithme RSA, première mise en œuvre du concept de cryptographie asymétrique dont les fondements furent posés par Whitfield Diffie et Martin Hellman en 1976.

[4] : **Taher Elgamal** naît en 1955 en Égypte. Il obtient un bachelor of science de l'université du Caire en 1981, puis un master of science et enfin un doctorat de l'université Stanford. En 1985, il publie un article intitulé un cryptosystème à clef publique et un schéma de signature basé sur les logarithmes discrets connu désormais sous le nom d'algorithme ElGamal

*[5] : **Pascal Paillier** est un cryptologue français né le 9 octobre 1971 à Saint-Germain-en-Laye. Il est connu pour avoir développé le cryptosystème de Paillier en 1999. Après avoir obtenu son doctorat en informatique à l'Université Paris VI, il a travaillé dans plusieurs institutions de recherche et d'enseignement en France, notamment à l'Institut National de Recherche en Informatique et en Automatique (INRIA) et à l'École Normale Supérieure.*

Cryptosystème GM

[6] : **Shafi Goldwasser** est une cryptographe et informaticienne renommée née en 1958 à New York. Elle a obtenu son doctorat en informatique à l'Université de Californie à Berkeley en 1984. Elle est surtout connue pour ses travaux sur la complexité computationnelle, la théorie des probabilités et la cryptographie. Goldwasser a été la première femme à remporter le prix Turing, l'une des récompenses les plus prestigieuses en informatique, en 2012. Elle a co-inventé, avec Silvio Micali, le système de cryptographie à clé publique connu sous le nom de cryptosystème RSA, l'un des algorithmes de chiffrement les plus largement utilisés au monde.

[7] :.**Silvio Micali** :Silvio Micali est un informaticien et cryptographe né en 1954 en Italie. Il a obtenu son doctorat en informatique à l'Université de Californie à Berkeley en 1982. Micali est également connu pour ses travaux sur la théorie des nombres, la complexité algorithmique et la cryptographie. En collaboration avec Shafi Goldwasser, il a développé des travaux fondamentaux dans le domaine de la cryptographie à clé publique, y compris le développement du protocole de preuve de connaissance zero-knowledge et du cryptosystème à clé publique nommé après leurs initiales, le cryptosystème GM.

Remarque : Le cryptosystème BGV est un système de chiffrement homomorphe développé par Zvika Brakerski, Craig Gentry et Vinod Vaikuntanathan. Le nom "BGV" est dérivé des premières lettres de leurs noms de famille. Ce système a été présenté pour la première fois dans un article intitulé "Fully Homomorphic Encryption without Bootstrapping" publié en 2012.Voici une brève biographie des principaux contributeurs :

[8] :**Zvika Brakerski** : Il est un chercheur en cryptographie et informaticien israélien. Il a obtenu son doctorat à l'Université Weizmann, en Israël, sous la supervision de Shafi Goldwasser, une pionnière de la cryptographie moderne. Brakerski a apporté des contributions significatives à divers domaines de la cryptographie, notamment la sécurité des chiffrements homomorphes.

[9] : **Craig Gentry** : Il est un chercheur américain en cryptographie, surtout connu pour ses travaux sur les chiffrements homomorphes. Gentry est titulaire d'un doctorat du Massachusetts Institute of Technology (MIT). Son article de 2009 sur le chiffrement homomorphe entièrement fonctionnel a été une percée majeure dans le domaine de la cryptographie.

[10] : **Vinod Vaikuntanathan** : Il est un informaticien et cryptographe indien, connu pour ses recherches en cryptographie théorique et en sécurité informatique. Vaikuntanathan a obtenu son doctorat à l'Université de Californie à Berkeley. Ses travaux se concentrent sur les constructions de chiffrement homomorphe et les protocoles sécurisés.

[11] **: B. Chor and N. Gilboa**, “Computationally private information retrieval,” in Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pp. 304–313, 1997.

[12] : **G. Barbujani and V. Colonna, Human genome diversity** : frequently asked questions, Trends in Genetics, vol. 26, no. 7, pp. 285295, 2010.

A

Code source

Livres :

- **"Homomorphic Encryption: Theory, Practice, and Applications" par O. Albrecht, F. Dachman-Weinberger, E. Orsini, and H. Peeters**
- **"Secure Multi-Party Computation" par D. Dolev and A. C. Yao**

Sites web :

CNIL - Chiffrement homomorphe : définition et avantages : https://www.cnil.fr/fr/definition/chiffrement-homomorphe

LePont - Le chiffrement homomorphe : définition et avantages : https://www.lepont-learning.com/fr/chiffrement-homomorphe-definition-avantages/

Statistique Canada - Technologies liées à la protection de la vie privée partie deux : introduction au chiffrement homomorphe : https://www.statcan.gc.ca/fr/science-donnees/reseau/chiffrement-homomorphe

Articles scientifiques :

"Homomorphic Encryption: A Tutorial" par D. Gentry, C. Gentry, S. Halevi, and V. Raykov : https://eprint.iacr.org/2010/271

"Somewhat Homomorphic Encryption" par D. Gentry, C. Gentry, S. Halevi, and V. Raykov : https://eprint.iacr.org/2008/068

"Fully Homomorphic Encryption: A Survey" par D. Boneh, E. Demaine, and M. Goodrich : https://eprint.iacr.org/2010/143

Cours en ligne :

"Homomorphic Encryption: A Gentle Introduction" par Dan Boneh (Stanford University) : https://www.coursera.org/learn/crypto

"Introduction to Homomorphic Encryption" par Nava Niv (Technion - Israel Institute of Technology) : https://www.edx.org/learn/computer-programming/raspberry-pi-foundation-introduction-to-encryption-and-cryptography

"Homomorphic Encryption and Applications" par Phong Q. Nguyen (University of Bristol) : https://www.coursera.org/learn/crypto