Résumé & Simplification du concept FHE

Thomas

September 2017

1 Notation

On suppose une donnée d que l'on souhaite chiffrer tel qu'on obtienne $\mathcal{C}(d)$ et que l'opération de déchiffrement soit $\mathcal{C}'$ tel que :

$$\mathcal{C}'(\mathcal{C}(d)) = d$$

Pour pouvoir exécuter toutes sortes de calculs sans décrypter les données il suffit qu'il existe une opération $*_1$ et $*_2$ tel que :

$$\mathcal{C}(d_1 + d_2) = \mathcal{C}(d_1) *_1 \mathcal{C}(d_2) \quad (1)$$
$$\mathcal{C}(d_1 \times d_2) = \mathcal{C}(d_1) *_2 \mathcal{C}(d_2) \quad (2)$$

Afin de réaliser les calculs $(+,\times)$ sur les données initiales il suffit de réaliser les opérations $(*_1,*_2)$ sur les données cryptées.

Exemple :
On souhaite calculer $d_1 + d_2 \times d_3$:

Il suffit de calculer $R = \mathcal{C}(d_1) *_1 (\mathcal{C}(d_2) *_2 \mathcal{C}(d_3))$*, puis de déchiffrer le résultat tel que :*

$$\mathcal{C}'(R) = \mathcal{C}'(\mathcal{C}(d_1) *_1 (\mathcal{C}(d_2) *_2 \mathcal{C}(d_3)))$$

$$\mathcal{C}'(R) = \mathcal{C}'(\mathcal{C}(d_1)) + \mathcal{C}'(\mathcal{C}(d_2) *_2 \mathcal{C}(d_3))$$

$$\mathcal{C}'(R) = d_1 + \mathcal{C}'(\mathcal{C}(d_2)) \times \mathcal{C}'(\mathcal{C}(d_3)) = d_1 + d_2 \times d_3$$

On dit qu'une méthode de cryptage est **pleinement homomorphique** si elle vérifie (entre autre) les propritétés (1) et (2).

2 Cryptage pleinement homomorphique

2.1 Chiffrage et opérations

Cette méthode, proposée par C. Gentry, définit un système qui chiffre les bits afin de pouvoir exécuter un nombre quelconque d'opérations logiques de type XOR et ET sur ces bits chiffrés.
On définit ici comme clef secrète ($\mathcal{C}$ dans la partie précédente) un long nombre entier **impair** p. Par ailleurs on peut définir une sorte de "multiple" de p, un nombre sous la forme $qp + e$, avec e un "petit" nombre correspondant au bruit et q un très grand nombre (i.e $q \gg p$). Il est alors impossible de le différencier d'un nombre quelconque, cependant celui qui connaît p peut facilement retrouver q en divisant le nombre précédent par p.

On peut donc chiffrer un bit b par : $\mathcal{C}(b) = pq + b + 2r$ où r représente le bruit. On obtient un système de chiffrement homomorphe pour le XOR et le ET, soit pleinement homomorphe.

Preuve : Soient b_1 et $_2$ deux bits chiffrés respectivement tels que :

$$\mathcal{C}(b_1) = c_1 = q_1 p + 2r_1 + b_1$$
$$\mathcal{C}(b_2) = c_2 = q_2 p + 2r_2 + b_2$$

Où : $r_1 \sim r_2 \ll p \ll q_1 \sim q_2$

En connaissant la valeur de p on peut calculer le reste de la division par p de :

$$c_1 + c_2 = (q_1 + q_2)p + 2(r_1 + r_2) + (b_1 + b_2)$$

On obtient alors : $2(r_1 + r_2) + (b_1 + b_2)$, or $2(r_1 + r_2)$ est pair donc il n'influe pas sur la parité du reste, et ainsi on en déduit que si le reste est impair cela signifie que b_1 est impaire ET b_2 pair (et inversement). On remarque donc que :

$$\mathcal{C}(b_1) + \mathcal{C}(b_2) = b_1 \oplus b_2$$

En calculant le reste de la division par p de :

$$c_1 c_1 = (q_1 q_2 p + 2q_1 r_2 + q_1 b_2 + 2q_2 r_1 + q_2 b_1)p + 2(2r_1 r_2 + r_1 b_2 + r_2 b_1) + b_1 b_2$$

On obtient alors $2(2r_1 r_2 + r_1 b_2 + r_2 b_1) + b_1 b_2$, dont la parité ne dépend que de $b_1 b_2$, qui est impair si et seulement si b_1 ET b_2 le sont. On remarque donc que :

$$\mathcal{C}(b_1) \times \mathcal{C}(b_2) = b_1 \wedge b_2$$

2.2 Nettoyage périodique du bruit

On remarque qu'au bout d'un certain nombre d'opération (en particulier les ET) il est possible d'atteindre un niveau de bruit suffisamment important (de l'ordre de p) tel qu'il deviendrait impossible de pouvoir déchiffrer les données puisque le reste de la division serait altéré.
L'idée est donc de nettoyer régulièrement le bruit issue des différentes opérations lorsque celui-ci devient trop important afin de pouvoir continuer les calculs. Pour ce faire on pourrait simplement déchiffrer le résultat partiel (en divisant par p) avant qu'il ne soit trop brouillé puis le chiffrer de nouveau avec un bruit initial. Cependant cela nécessite la connaissance de la clef secrète (p) par l'opérateur de calcul.

Une astuce conjointe à cette méthode proposée par C. Gentry consiste à "simplement" effectuer le déchiffrement via une division de manière homomorphique, afin de ne jamais avoir à divulguer la clef privée p. Cette phase est alors appelée **bootstrapping**.
Pour que cette étape soit un succès il est néanmoins nécessaire que les opérations induites par le déchiffrement homomorphique soit réalisables sans que le bruit généré ne soit trop important. C'est pourquoi les nombres p et q_i se doivent d'être très grands (avec toujours $q_i \gg p$).

3 Autres chiffrements homomorphiques

3.1 Leveled Homomorphic Encryption

Les paramètres de chiffrement permettent de définir un nombre pré-déterminé d'opérations sur les données sans qu'elles ne devienne indéchiffrable. Il est donc possible de se passer de l'étape détaillée précédemment consistant à nettoyer le bruit régulièrement.

cf. DOWNLIN - CryptoNets report

3.2 Practical Homomorphic Encryption

Il s'agit de l'utilisation d'une LHE en sus d'un encodage particulier des données afin d'optimiser leurs traitements au cours des différentes étapes de l'algorithme employé.
Bien que cette méthode soit plus restrictive que le chiffrement pleinement homomorphique, elle présente néanmoins l'avantage de proposer un gain en performance substantiel.