

- de vous transmettre le résultat. Puis vous divisez le résultat par C : vous avez votre moyenne, car la moyenne de $Cd_1, Cd_2, \dots, Cd_n$ est C fois la moyenne de $d_1, d_2, \dots, d_n$.

Avec ce procédé, vous pouvez faire exécuter par l'opérateur extérieur d'autres types de calculs. Pour connaître le nombre de données qui dépassent un certain nombre B , vous lui demanderez le nombre de données qui dépassent CB . Le tri de vos données par ordre croissant est aussi faisable, puisque si C est positif, le tri de $Cd_1, Cd_2, \dots, Cd_n$ exige les mêmes permutations que $d_1, d_2, \dots, d_n$.

Toutes les moyennes pondérées, par exemple la moyenne $(d_1 + 2d_2 + 5d_3 + d_4)/9$, sont aussi possibles sans dévoiler C , car le résultat sur les données multipliées par C est égal à C fois le résultat attendu. Certaines opérations un peu plus compliquées sont envisageables: pour le calcul de $d_1d_2 + d_3d_4 + d_5d_6$, il suffira de diviser le résultat que $\mathcal{O}$ fournit par C^2 .

Cependant, vous ne pourrez pas demander $d_1 + d_2d_3$ et, finalement, ce n'est qu'un faible sous-ensemble de calculs que vous saurez faire exécuter à l'extérieur sans avoir à divulguer vos secrets. En particulier, pas question de faire opérer des algorithmes complexes sur vos données, comme l'identification, parmi les nombres confiés, de ceux qui sont des nombres premiers.

Remarquons aussi que le chiffrement de vos données en les multipliant par une constante C laisse transparente certaines informations que vous ne souhaitez peut-être pas que l'opérateur $\mathcal{O}$ connaisse: le numéro de la donnée la plus grande ou la plus petite, les valeurs égales, les données classées, etc. Le problème sera résolu si vous disposez

d'une méthode de chiffrement permettant à la fois de bien masquer toute l'information et autorisant l'exécution d'additions et de multiplications sans rien révéler: avec ces deux opérations, on reconstitue tout calcul.

DES CHIFFREMENTS PLEINEMENT HOMOMORPHES

Notons aussi qu'il n'est pas essentiel que l'opération effectuée par $\mathcal{O}$ soit identique à celle que vous voulez au final. En effet, supposons que le chiffrement de la donnée d soit $\mathcal{C}(d)$ et que le déchiffrement (que vous seul savez effectuer) soit l'opération $\mathcal{C}': \mathcal{C}'(\mathcal{C}(d)) = d$. Pour réussir à faire exécuter toutes sortes de calculs par un opérateur extérieur, il suffirait qu'il existe une opération $*_1$ et une opération $*_2$ (pas nécessairement l'addition et la multiplication habituelles) vérifiant:

$$\begin{aligned}\mathcal{C}(d_1 + d_2) &= \mathcal{C}(d_1) *_1 \mathcal{C}(d_2) \quad (\text{propriété 1}) \\ \mathcal{C}(d_1 \times d_2) &= \mathcal{C}(d_1) *_2 \mathcal{C}(d_2) \quad (\text{propriété 2}).\end{aligned}$$

En exécutant $*_1$ et $*_2$ à la place de $+$ et $\times$, l'opérateur 2 produira à partir des données chiffrées un résultat qui, après le déchiffrement par $\mathcal{C}'$, sera exactement ce que vous voulez.

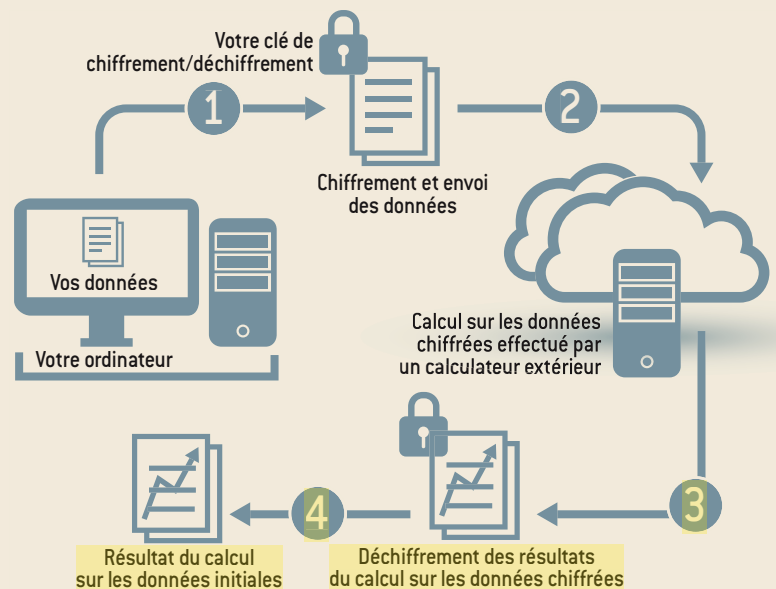
Par exemple, si vous souhaitez connaître $d_1 + d_2d_3$ sans le calculer vous-même (ce qui est impossible avec la méthode de multiplication par C), vous enverrez vos données chiffrées $\mathcal{C}(d_1), \mathcal{C}(d_2), \mathcal{C}(d_3)$ à l'opérateur en lui demandant de calculer $\mathcal{C}(d_1) *_1 (\mathcal{C}(d_2) *_2 \mathcal{C}(d_3))$ et de vous renvoyer le résultat R . Grâce à l'opération $\mathcal{C}'$, vous obtiendrez alors la valeur de $d_1 + d_2d_3$, puisque:

$$\begin{aligned}\mathcal{C}'(R) &= \mathcal{C}'(\mathcal{C}(d_1) *_1 (\mathcal{C}(d_2) *_2 \mathcal{C}(d_3))) = \\ &= \mathcal{C}'(\mathcal{C}(d_1)) + \mathcal{C}'(\mathcal{C}(d_2) *_2 \mathcal{C}(d_3)) = \\ &= d_1 + \mathcal{C}'(\mathcal{C}(d_2) \times \mathcal{C}(d_3)) = d_1 + d_2d_3.\end{aligned}$$

La méthode fonctionnera pour toute

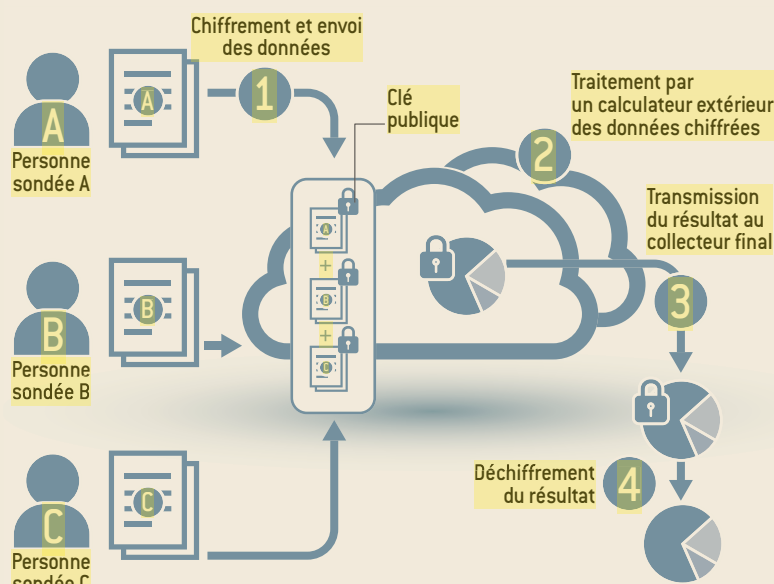
CALCUL DÉLOCALISÉ

Avec les chiffrements dits homomorphes, un opérateur calcule sur des données chiffrées dont il n'a pas connaissance. Une fois le calcul achevé, il ne peut pas connaître le résultat trouvé, que seul le commanditaire du calcul déchiffre. Vous chiffrez vos données avec une clé secrète et vous les envoyez à l'opérateur extérieur (1). L'opérateur effectue des calculs avec les données chiffrées et vous envoie les résultats (chiffrés) sans avoir eu connaissance de la clé (2, 3). Vous déchiffrez les résultats reçus grâce à la clé secrète (4): vous avez ainsi connaissance des résultats d'un calcul que vous n'avez pas fait, résultats auxquels l'opérateur n'a pas pu avoir accès. Pour des calculs simples, des séries d'additions et de soustractions, la mise au point d'applications réelles telles que les systèmes de vote est possible.



PRÉSERVER LA CONFIDENTIALITÉ

Imaginons que l'on souhaite établir une statistique sur un sujet délicat en utilisant Internet et en garantissant à chacune des personnes sondées que les données individuelles contenues dans les réponses ne seront pas divulguées. La procédure est la suivante. Chaque personne chiffre ses réponses avec la clé publique du collecteur final et les envoie à un calculateur extérieur (1). Ce calculateur traite ces données chiffrées et envoie le résultat au collecteur final (2, 3). Ce dernier déchiffre le résultat et obtient la statistique souhaitée (4). Le calculateur extérieur ne peut pas accéder aux informations privées des sujets de l'enquête, car il ne dispose pas de la clé privée de déchiffrement, et le collecteur final ne peut pas accéder aux données individuelles chiffrées individuelles transmises.



expression, aussi compliquée soit-elle, constituée d'additions et de multiplications. Puisque dès qu'on sait mener des additions et des multiplications de cette façon, on peut exécuter tout algorithme, le problème des calculs secrets par un tiers se ramène à celui de trouver $\mathcal{E}$, $\mathcal{D}$, $\ast_1$, $\ast_2$, vérifiant les propriétés 1 et 2.

Les méthodes de chiffrement vérifiant les propriétés 1 et 2 sont dites «pleinement homomorphes». Le terme vient du langage mathématique où les homomorphismes sont des fonctions qui transforment une opération en une autre, comme l'indique par exemple la propriété 1.

Ces homomorphismes jouent un rôle important en mathématiques et chacun connaît le logarithme qui vérifie $\log(a \times b) = \log(a) + \log(b)$ et qui transforme donc des multiplications en additions (ici, $\ast_2$ est l'addition habituelle). Avant l'avènement des ordinateurs, c'était le moyen le plus efficace pour mener à la main des calculs numériques, et c'est pourquoi on dressait des tables de logarithmes contenant des millions de données.

En mathématiques, on connaît de nombreux homomorphismes, mais aucun des homomorphismes classiques ne peut servir à concevoir un chiffrement $\mathcal{E}$ homomorphe car l'opération de déchiffrement $\mathcal{D}$ est alors trop simple; or une bonne méthode cryptographique est une méthode où il est difficile de trouver $\mathcal{D}$. Le problème des systèmes de chiffrement pleinement homomorphes n'était donc pas évident au moment où, en 1978, l'idée en a été envisagée par les Américains Ron Rivest et Leonard Adleman et le Grec Michael Dertouzos. On s'est longtemps demandé si de

telles méthodes existaient. Le problème a parfois été qualifié de graal de la cryptographie.

UNE PERCÉE DÉCISIVE

Une percée décisive a été réalisée en 2009 par Craig Gentry, de l'université Stanford et aujourd'hui au centre Thomas Watson d'IBM à New York. Craig Gentry a proposé une méthode qui résout théoriquement le problème et a pour cela reçu une bourse de 625 000 dollars de la fondation MacArthur. La méthode qu'il a initialement proposée n'est pas applicable, car elle exige des calculs trop volumineux au moment du chiffrement (la transformation des données avant de les envoyer) et du déchiffrement. De plus, la taille des données que doit manipuler l'opérateur extérieur $\mathcal{D}$, et donc la quantité de calculs qu'il doit mener, est des millions de fois celle nécessaire quand les données ne sont pas chiffrées.

Depuis 2009, une multitude de recherches ont été faites pour trouver des schémas plus économiques. En quelques années, les progrès ont rendu envisageables certaines applications, et l'espoir de méthodes encore plus économiques et faciles à mettre en œuvre suscite de nombreux travaux parmi les spécialistes.

Avant de donner des précisions sur les idées de la percée décisive de Craig Gentry, revenons sur les applications des chiffrements pleinement homomorphes ou même partiellement homomorphes.

L'algorithme RSA (conçu en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman) est une méthode de chiffrement à clé publique: pour chiffrer un message, on utilise la clé publique de

>