Cette dernière séquence présente les techniques de calcul distribué respectueuses de la vie privée et qui peuvent servir à calculer des résultats obtenus à partir des données personnelles de grand nombre d'individus.

Quels sont les traitements globaux que l'on souhaite réaliser ?
Du **traitement à grande échelle**, comme des calculs statistiques par exemple pour réaliser des études épidémiologiques, de la fouille de données pour classifier, calculer des profils ou tout type de traitements Big Data basés sur les techniques map reduce et bien d'autres calculs encore.
Les **hypothèses** que l'on fait dans cette séquence sont :

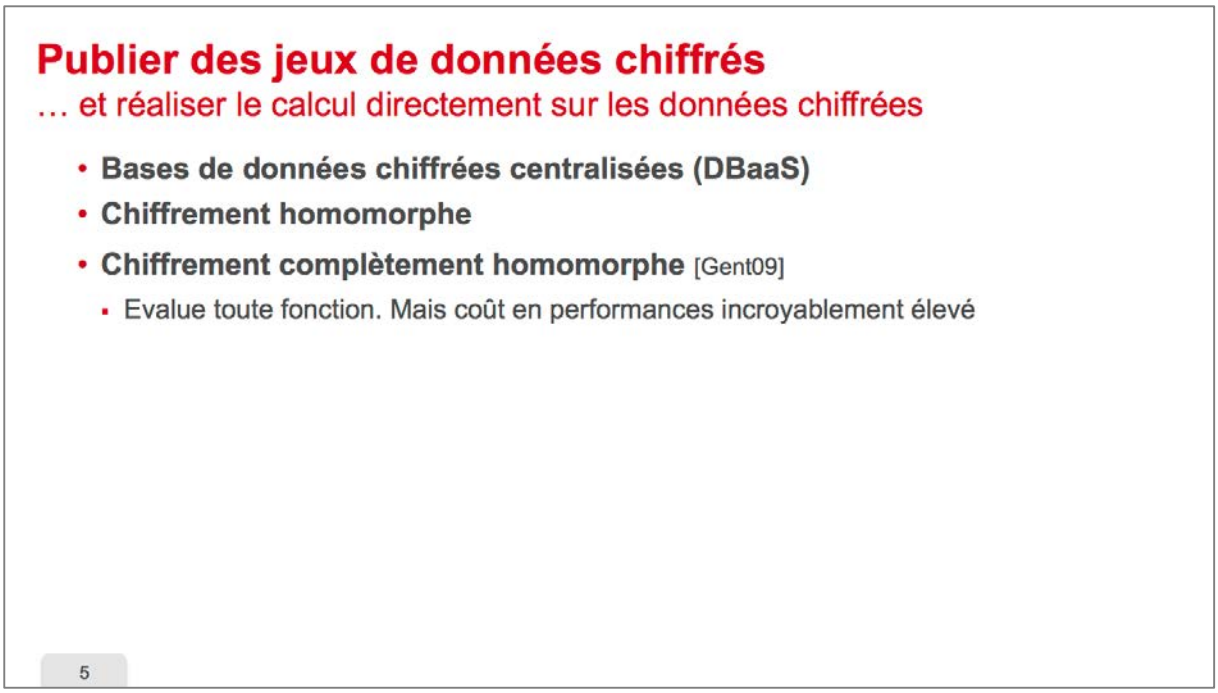
- les données de base du calcul sont détenues par les individus
- seul le résultat du calcul doit être révélé.
- L'infrastructure extérieure qui peut servir au calcul ou qui sert aux échanges de données entre les participants ne doit pas avoir accès aux données de base qui permettent de calculer le résultat.

Les **participants et l'infrastructure** peuvent donc être considérés comme étant :

- honnêtes, c'est-à-dire qu'ils suivent le protocole,
- semi-honnête, ils suivent aussi le protocole mais cette fois, ils tentent d'inférer de l'information
- malveillants et dans ce cas-là, ils ne suivent pas le protocole et essayent essentiellement d'inférer de l'information.

La première solution qu'on peut imaginer, c'est d'utiliser du chiffrement.
Chaque participant va chiffrer ses données et va les transmettre à destination d'une base de données centralisée qui va effectuer le calcul sur le chiffré.
La question qui se pose, c'est comment le serveur peut-il calculer le résultat si les données sont chiffrées et que bien sûr, le serveur ne dispose pas de la clé qui permet de les déchiffrer?

On peut utiliser du chiffrement homomorphe. C'est un mode de chiffrement qui **préserve certaines opérations** qui peuvent être réalisées directement sur les données chiffrées. Par exemple si on chiffre des données avec un algorithme RSA, on va préserver la multiplication de manière à ce que si on multiplie entre eux 2 nombres chiffrés, le résultat donne un nombre chiffré qui une fois déchiffré est le résultat de la multiplication sur les nombres en clair.
Mais peut-on réaliser de cette manière n'importe quel calcul ? Pas encore, du moins pas à l'heure actuelle parce qu'il faudrait du **chiffrement complètement homomorphe** qui préserve suffisamment d'opérations pour pouvoir implémenter n'importe quel traitement.

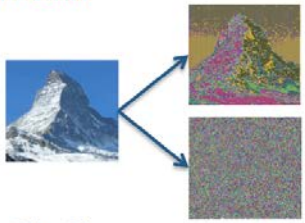


Il y a des résultats théoriques là-dessus, mais qui ne sont pas encore exploitables en pratique, car incroyablement coûteux, et en tout cas pas exploitables dans le cadre de la gestion de données.

Publier des jeux de données chiffrés

… et réaliser le calcul directement sur les données chiffrées

- **Bases de données chiffrées centralisées (DBaaS)**
- **Chiffrement homomorphe**
- **Chiffrement complètement homomorphe** [Gent09]
- **Utiliser du chiffrement déterministe**
  - Préserve l'égalité, l'ordre, etc. Ex. m1=m2 ⇔C(m1) = C(m2)
  - Inférences ?



6

Pour effectuer les opérations utiles au traitement base de données, il faudra donc utiliser des modes de chiffrement laissant la possibilité de réaliser des traitements de données quitte à dégrader la sécurité. Par exemple, un mode de chiffrement qui est largement utilisé pour préserver les traitements en bases de données est le chiffrement déterministe.

L'effet en est montré sur cette image :

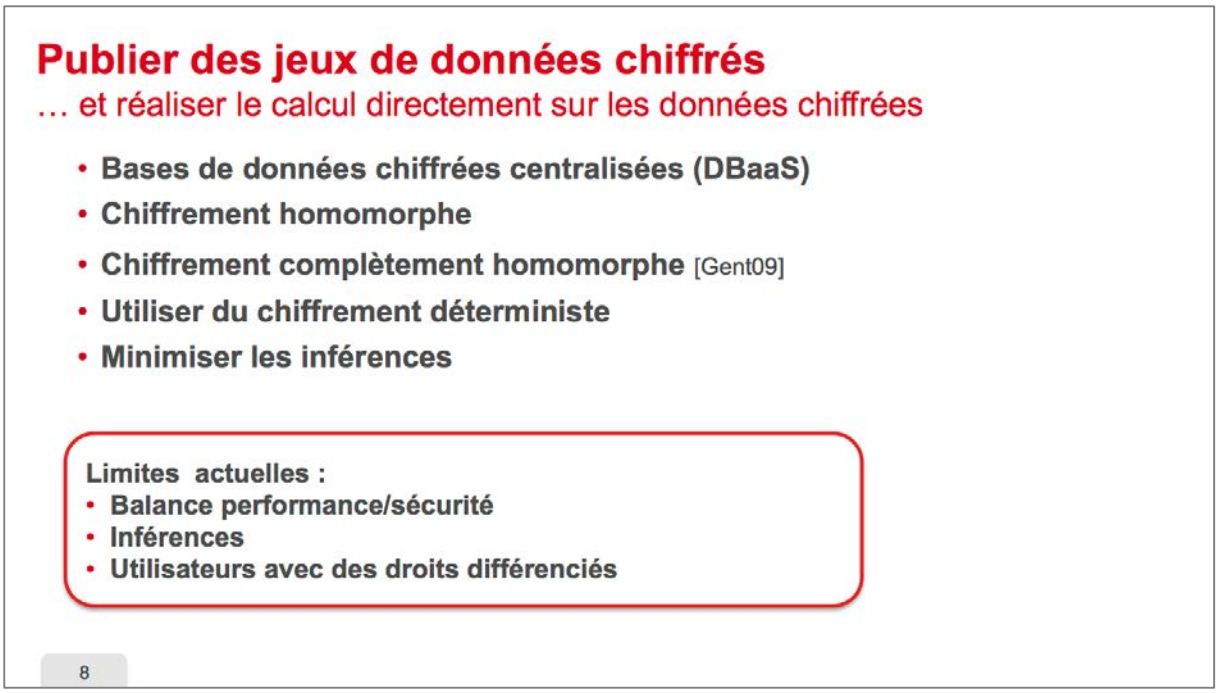
- dans celle du haut chaque pixel est chiffré en déterministe,
- dans celle du bas en non déterministe.

L'avantage avec l'image du haut est que les pixels restent indexables dans la base de données. Ainsi rechercher du bleu dans cette image revient à rechercher tous les pixels ayant pour valeur la valeur chiffrée de bleu.

Avec ce type de chiffrement, l'indexation est permise, mais la distribution de données est bien sûr visible.

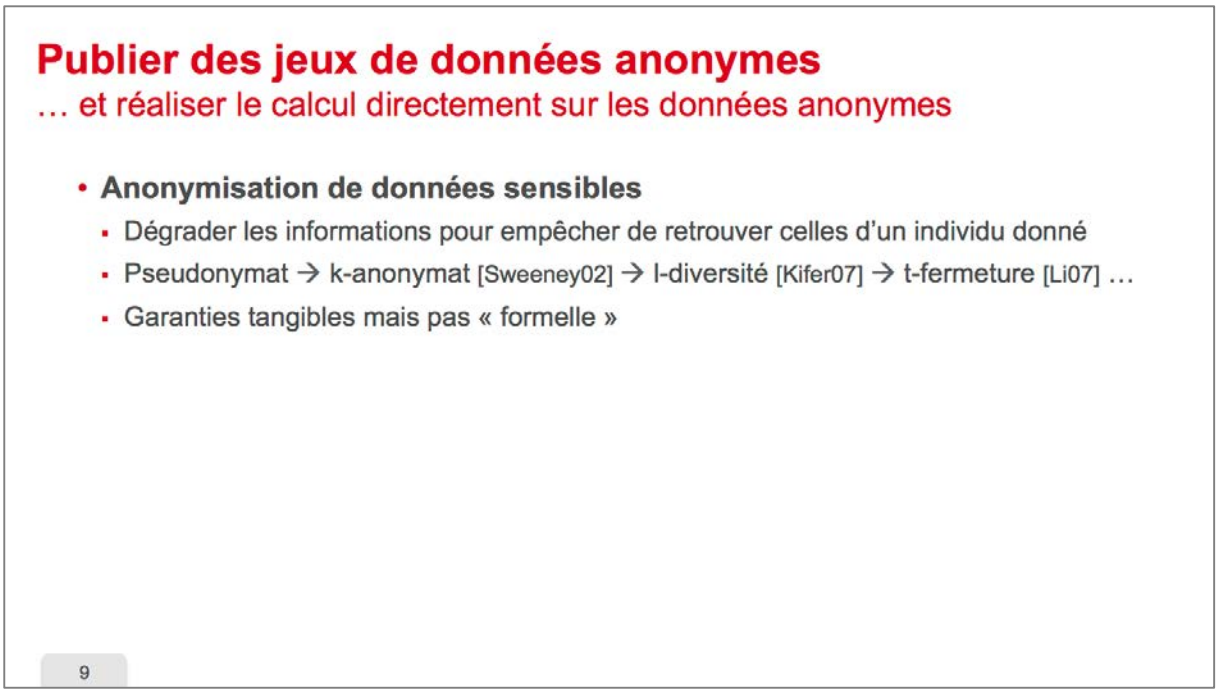
Si on connaissait cette distribution, il est facile de renverser le chiffrement.

Le problème est donc aussi de chiffrer mais en se protégeant des inférences. Certaines techniques permettent par exemple de **favoriser du chiffrement opaque** à chaque fois que c'est possible et d'autres permettent d'attacher aux données des **étiquettes qui sont statistiquement indistinguables** et qui permettent quand même au serveur d'évaluer certains calculs.



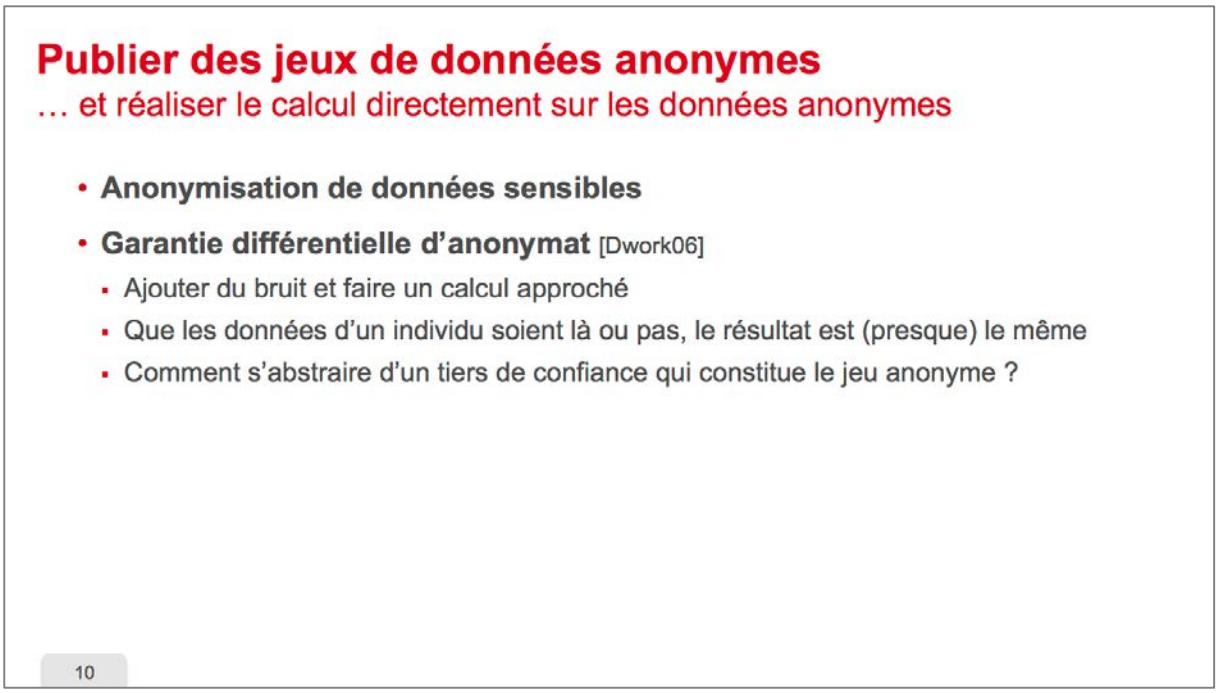
Ces techniques ont actuellement de nombreuses limites :

- le **compromis** fait sur la **sécurité** pour conserver de bonnes **performances** est fort, car les **inférences** sont possibles.
- Si les utilisateurs ont des droits différenciés, il faudra chiffrer les données avec des clés de chiffrement différentes, ce qui rend la gestion très complexe.

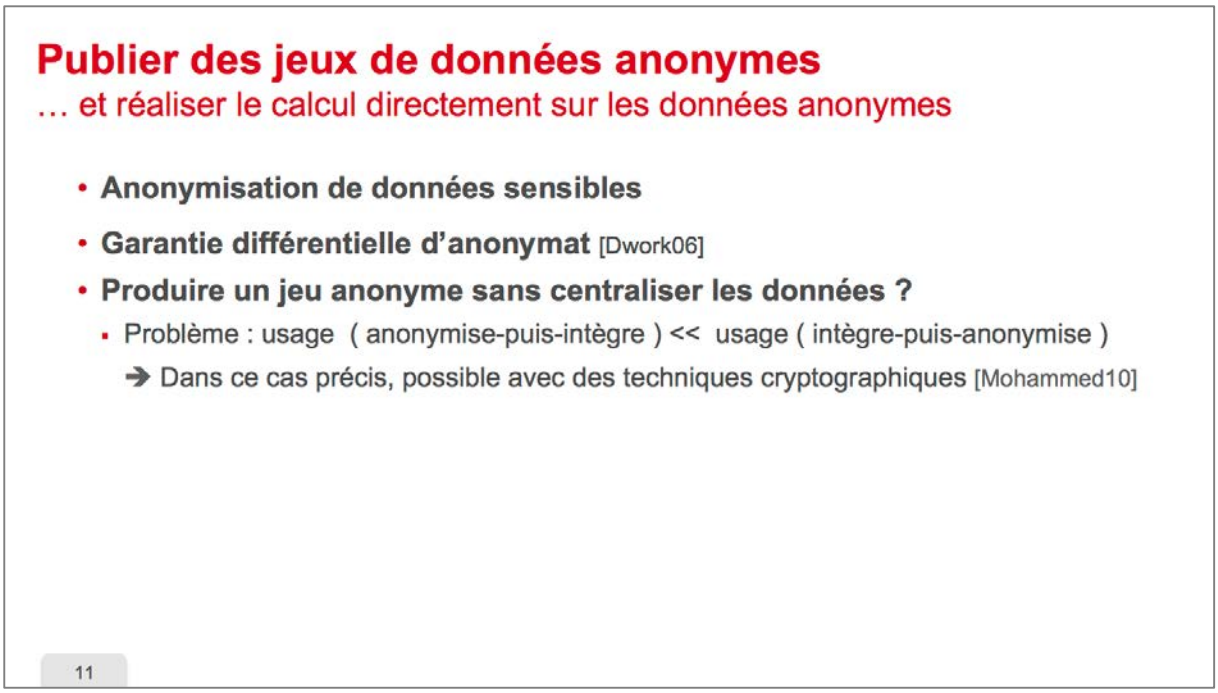


Une deuxième alternative consiste à anonymiser les données de chacun et à réaliser le calcul sur les données anonymes.

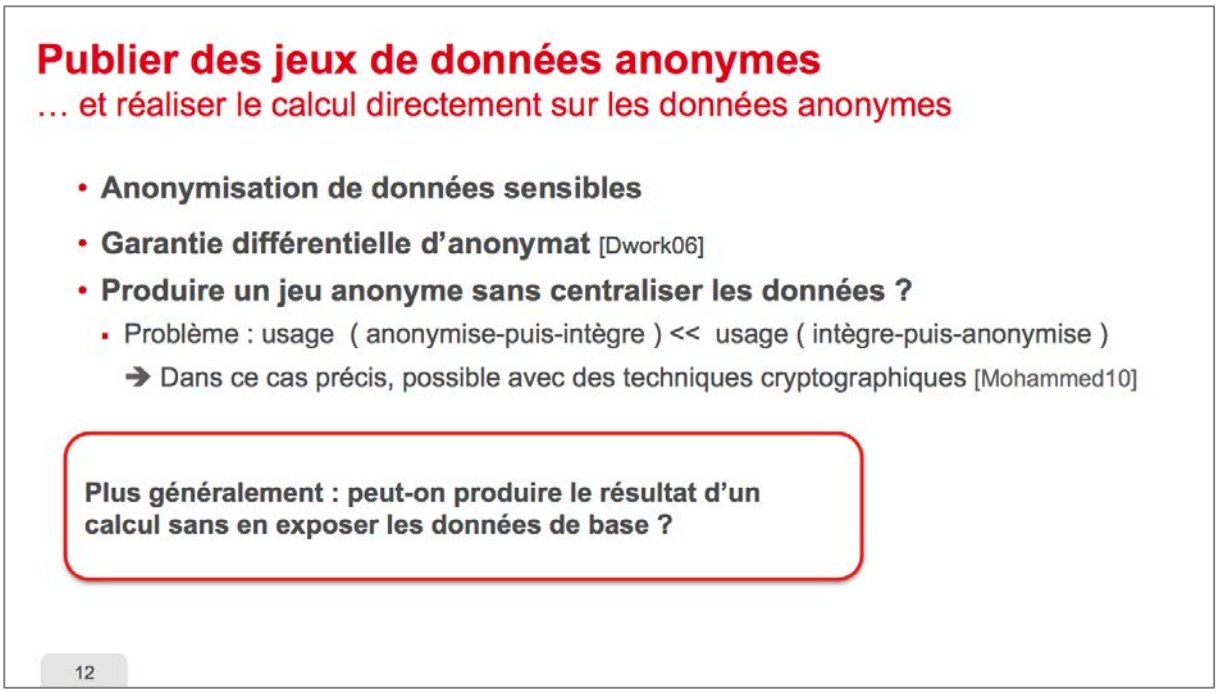
Une première solution pour obtenir l'anonymat est de généraliser ou supprimer certaines informations jusqu'à éviter de pouvoir identifier celles qui correspondent à un individu donné.
Mais plus le jeu de données devient anonyme et moins il est utilisable.
Un simple pseudonymat où les identifiants présents dans le jeu données sont remplacés par un pseudonyme ne suffit pas.
Le **k-anonymat** va plus loin et propose de généraliser aussi les éléments de jeu données qui sont perçus comme des quasi identifiants et qui pris en combinaison, pourraient identifier une personne en particulier.
La **l-diversité** propose de ne jamais associer une seule et même valeur sensible pour un même jeu de quasi identifiants.
La **t-fermeture** va encore plus loin en cherchant à respecter les distributions connues pour les données sensibles.



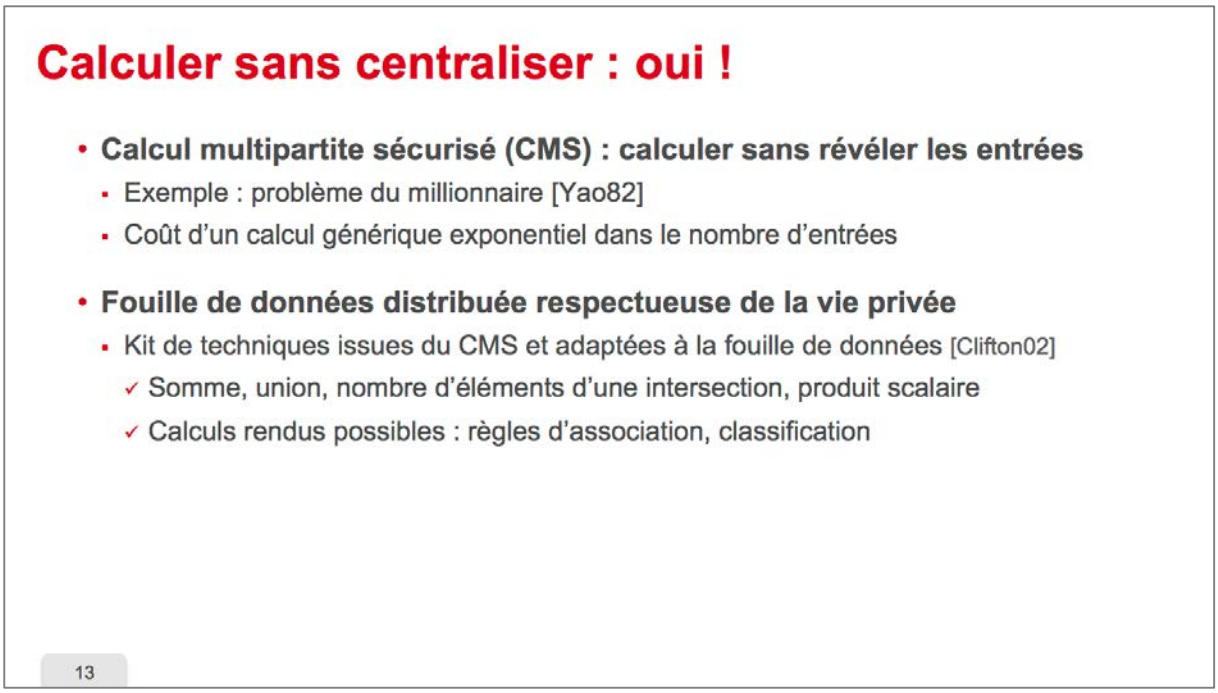
Une seconde technique consiste à introduire de fausses données de manière à masquer le fait qu'un individu appartient ou n'appartient pas au jeu de données.
Ainsi, un même calcul réalisé sur 2 jeux de données qui ne diffèrent que par les données d'un seul individu donnera le même résultat.
Les garanties offertes sont ainsi plus formelles qu'avec les techniques précédentes.
Mais un autre problème se pose: comment générer un tel jeu de données anonymes?



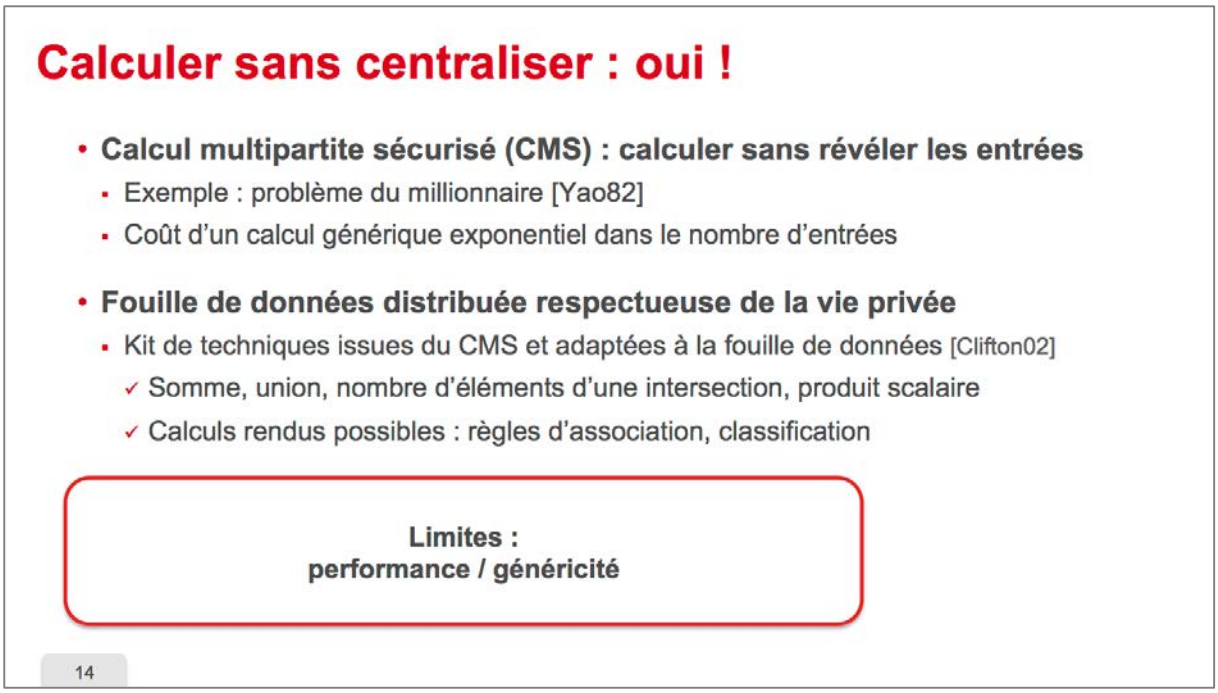
Si chaque individu anonymise lui-même ses données localement, l'usage qui pourra être fait des données sera très dégradé.
Pour anonymiser correctement, il faut confronter les données de l'ensemble de tous les individus pour produire le jeu anonyme



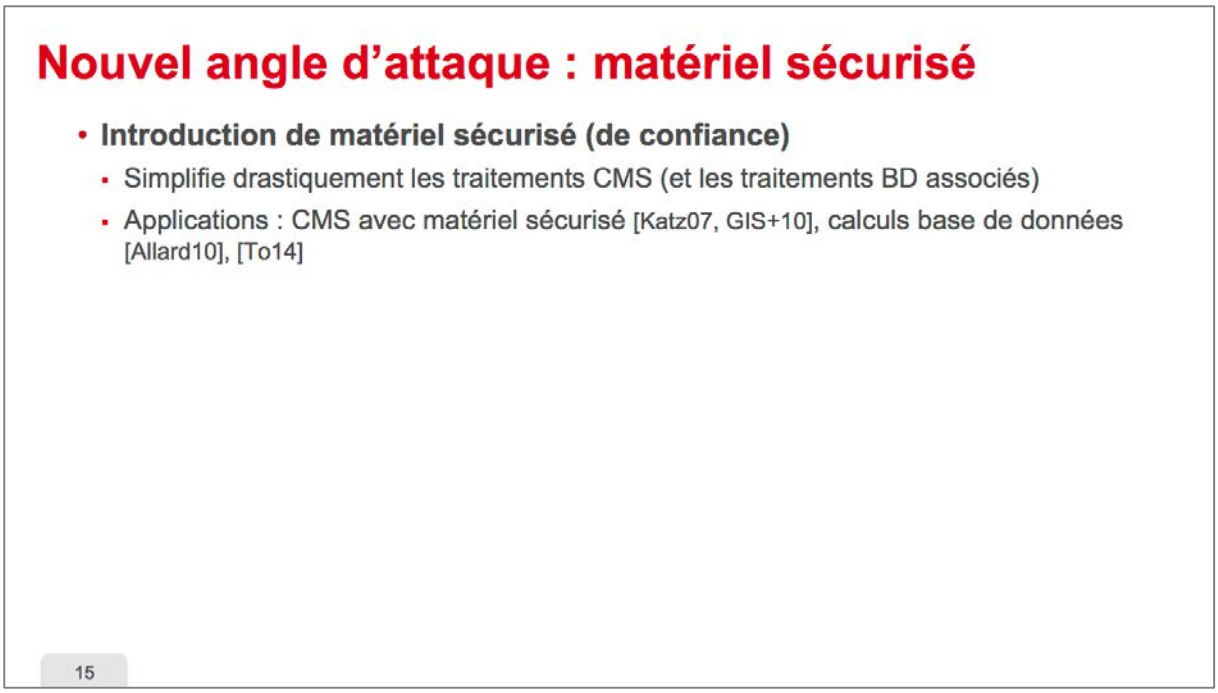
. Ca revient donc au problème présenté précédemment, celui de calculer un résultat sans révéler les données de base qui permettent le calcul.



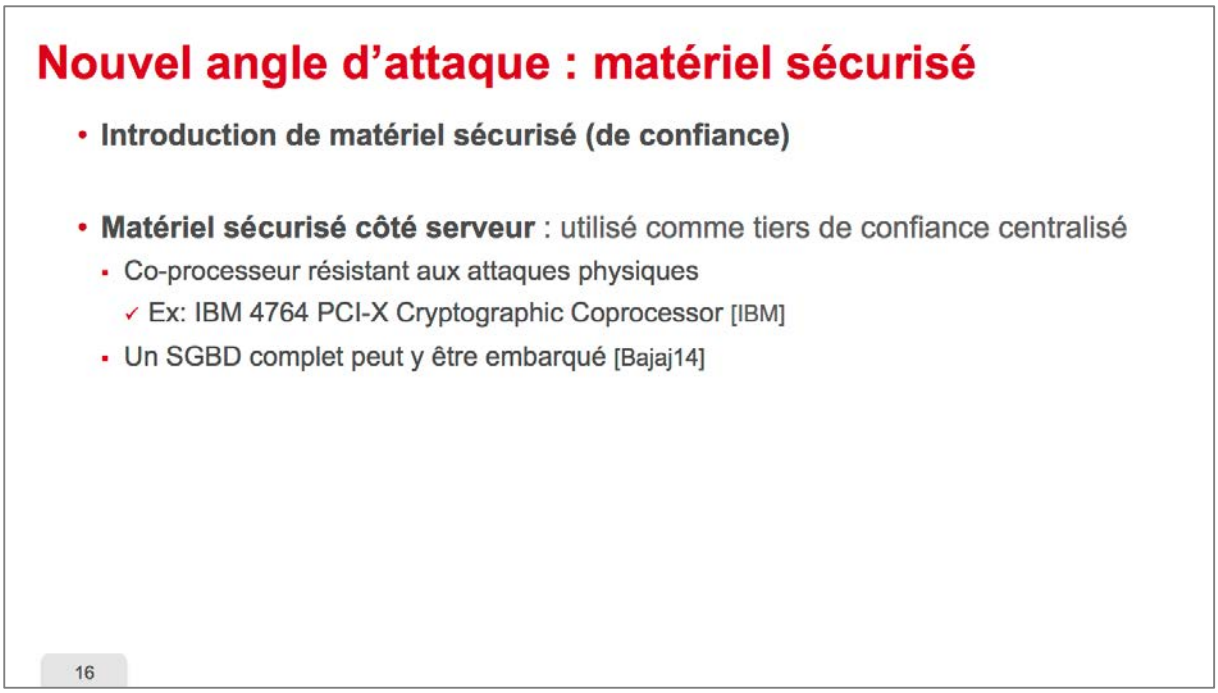
C'est un problème qui est étudié en cryptographie dans le cadre du **calcul multipartite sécurisé** (CMS). Mais le coût d'un calcul générique est exponentiel dans le nombre d'entrées.
Certaines **techniques de fouille de données** peuvent toutefois être dérivées de ces techniques de calcul multipartite sécurisé et **permettent de calculer des règles d'associations ou de faire de la classification de manière sécurisée**.



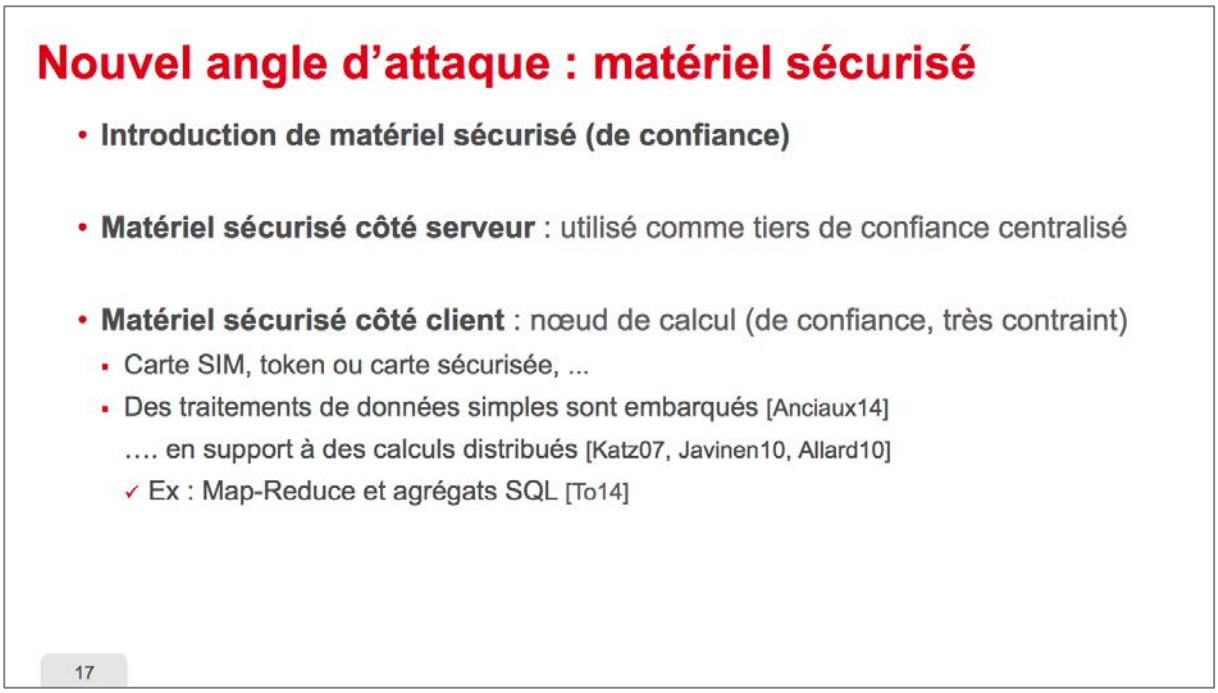
Mais à l'heure actuelle, ces techniques cryptographiques ne permettent pas de répondre à la plupart des problèmes de gestion de données.



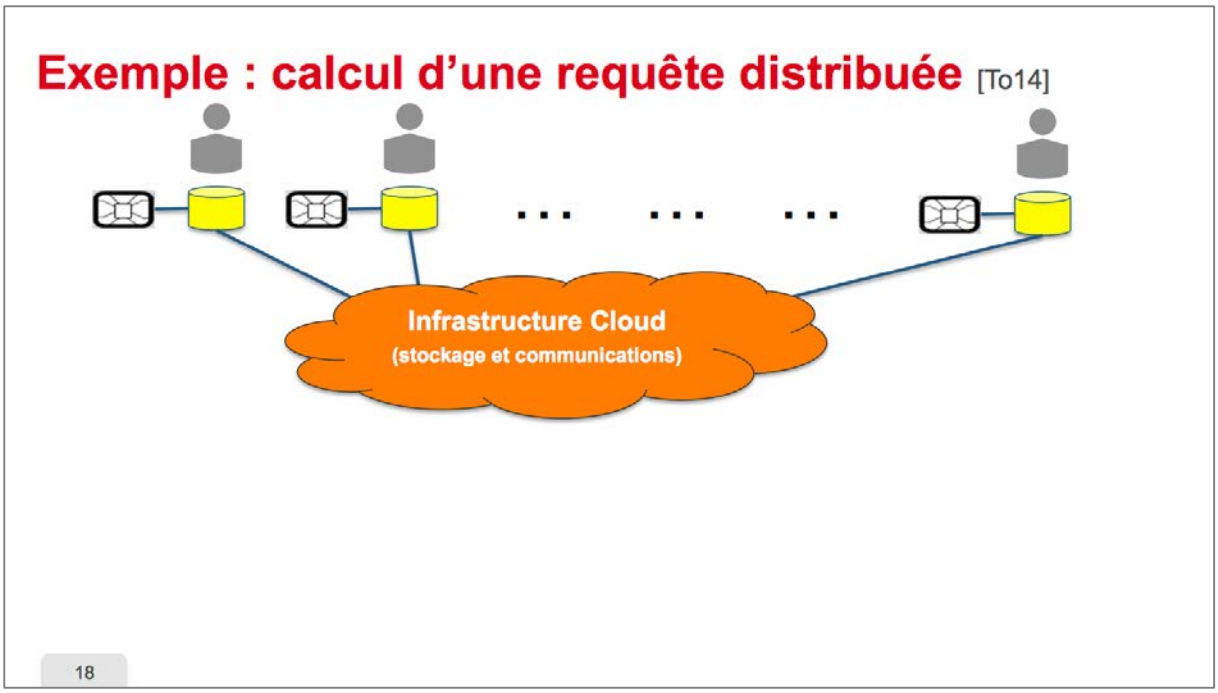
Un nouvel angle d'attaque émerge actuellement grâce à l'introduction de matériels sécurisés ou de matériels de confiance qui simplifient largement la problématique classique du calcul multipartite sécurisé.



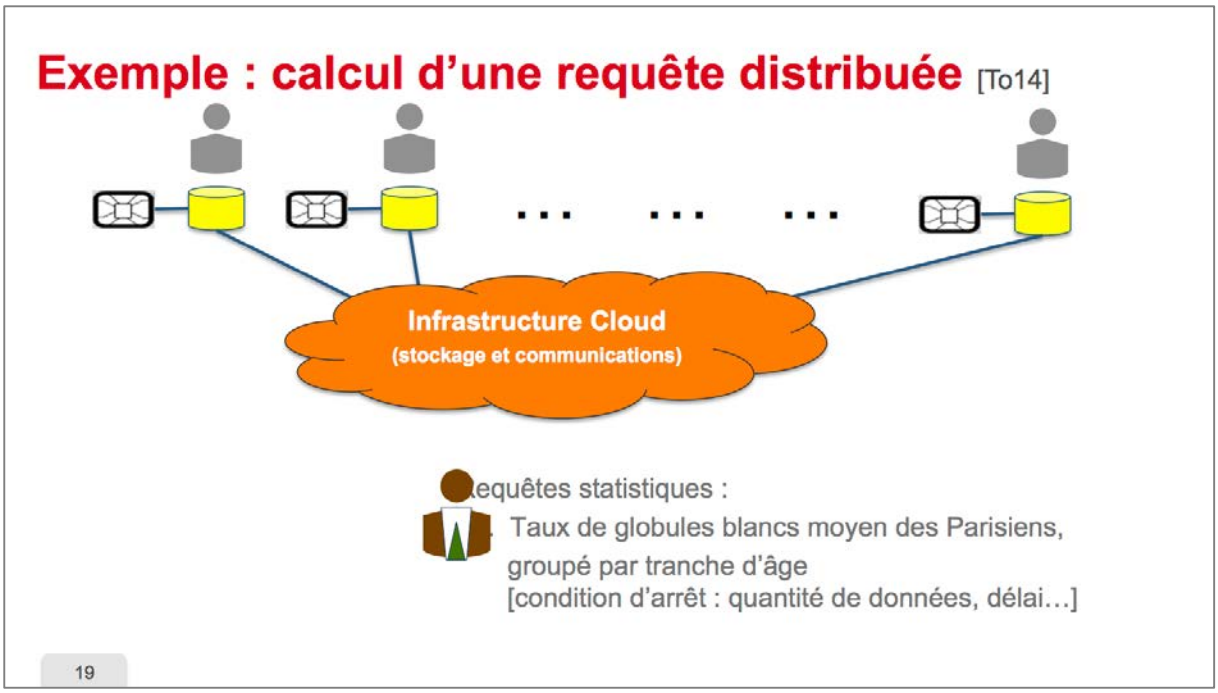
Par exemple un **co-processeur sécurisé** qui ne peut pas être observé peut être placé côté serveur pour collecter des données chiffrées - donc celles des participants - et les déchiffrer en interne pour calculer un résultat qui seul sera exposé. La solution Trusted DB a été proposée dans cette optique.



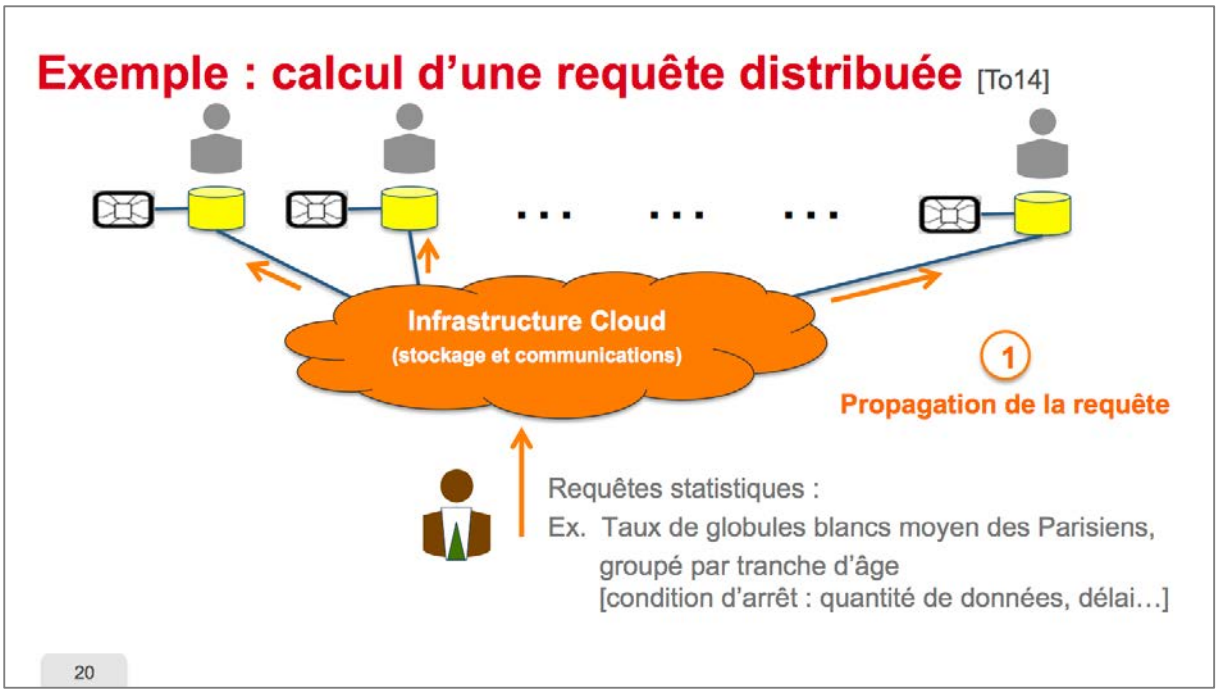
` Autre exemple, du matériel sécurisé peut être présent côté client, il peut s'agir d'une carte SIM embarquée dans un téléphone portable ou de tout autre objet sécurisé à disposition de l'individu. Comme nous l'avons vu lors de la troisième séquence de cette semaine, les traitements peuvent être embarqués dans ce type de dispositif.
Si les dispositifs peuvent s'échanger entre eux des données via une infrastructure, le tout peut être vu comme une plate-forme distribuée et sécurisée. À l'heure actuelle, des calculs d'agrégats SQL et des traitements Map reduce, par exemple, peuvent être réalisés de cette manière.



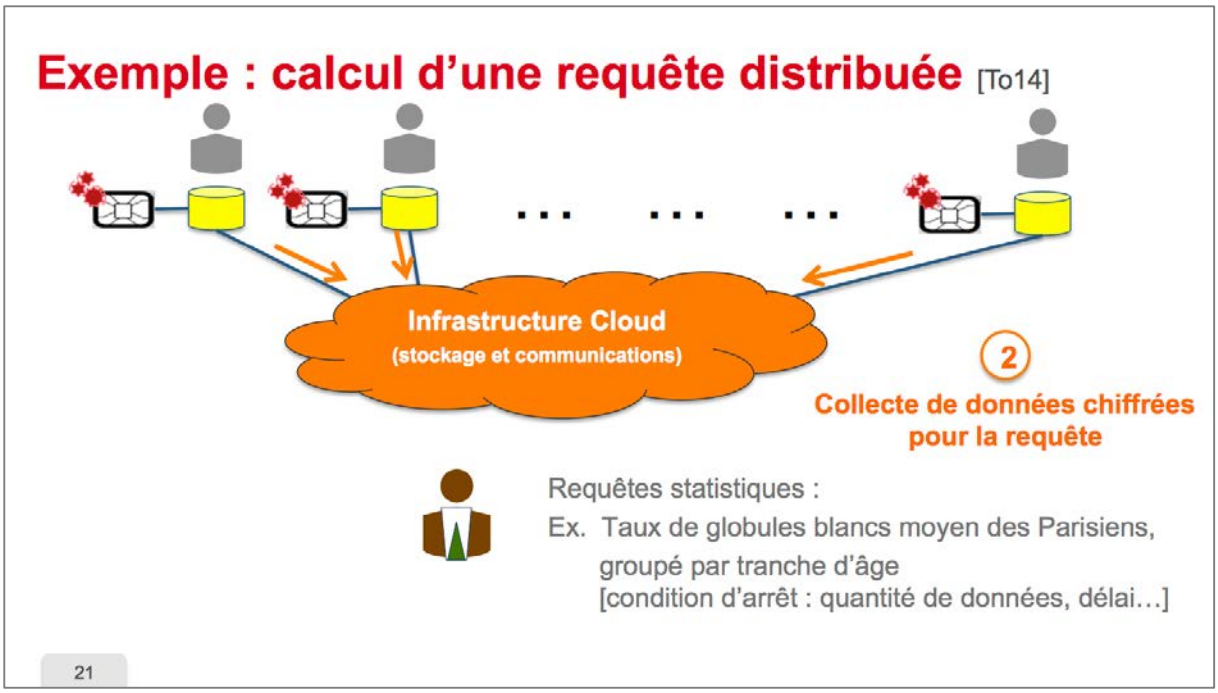
De façon très synthétique, voilà comment fonctionne ce type de calcul.
Supposons un grand nombre de citoyens disposant chacun de leurs données et d'un dispositif personnel sécurisé. Ces citoyens doivent être capables d'échanger des données au travers d'une infrastructure.



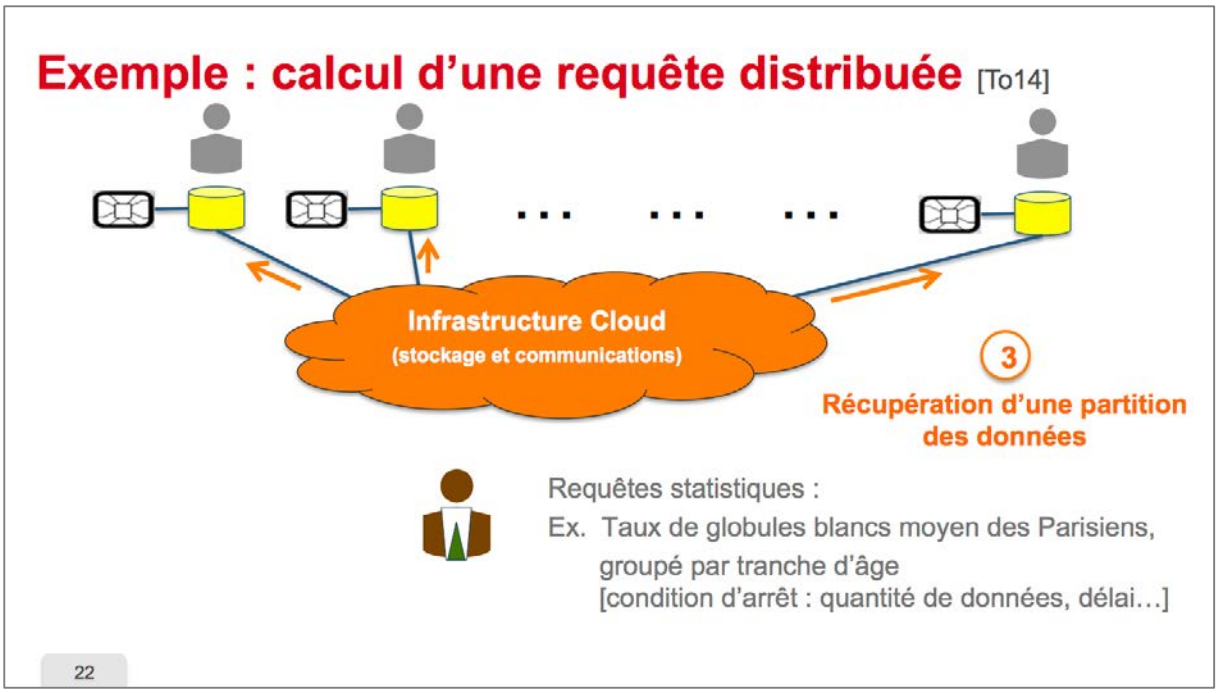
Supposons un individu tiers habilité à réaliser un calcul statistique.



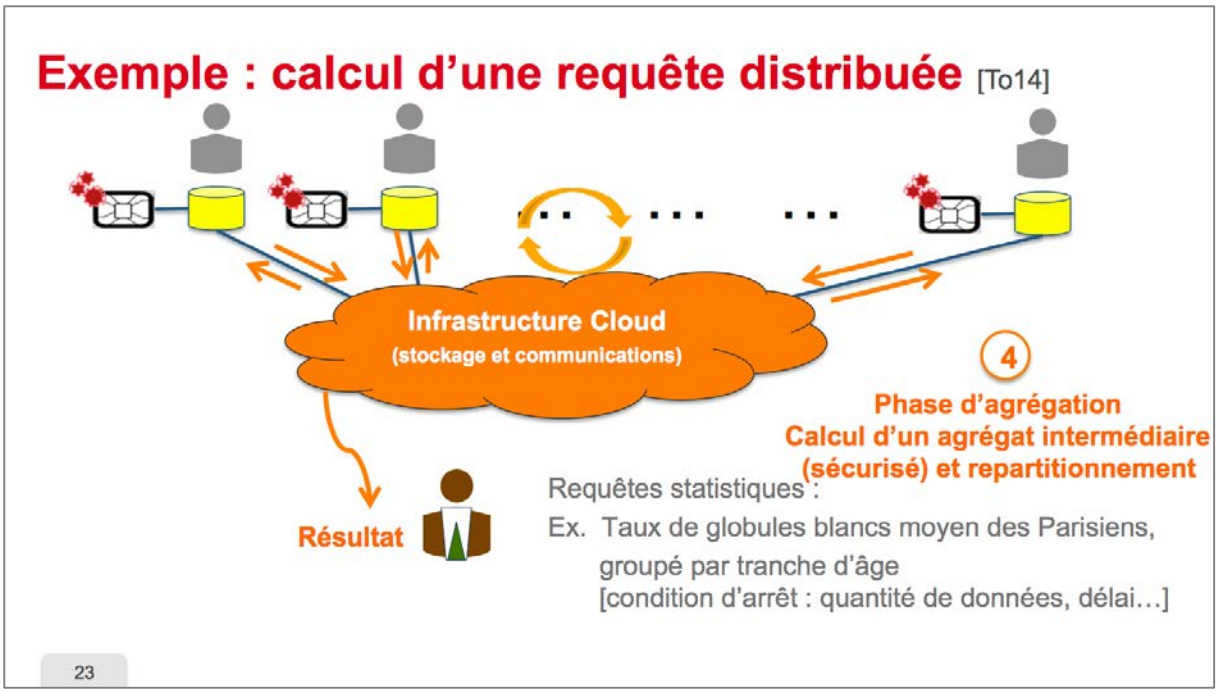
Dans une première phase, la requête que cet individu va poser est chiffrée et transmise aux participants.



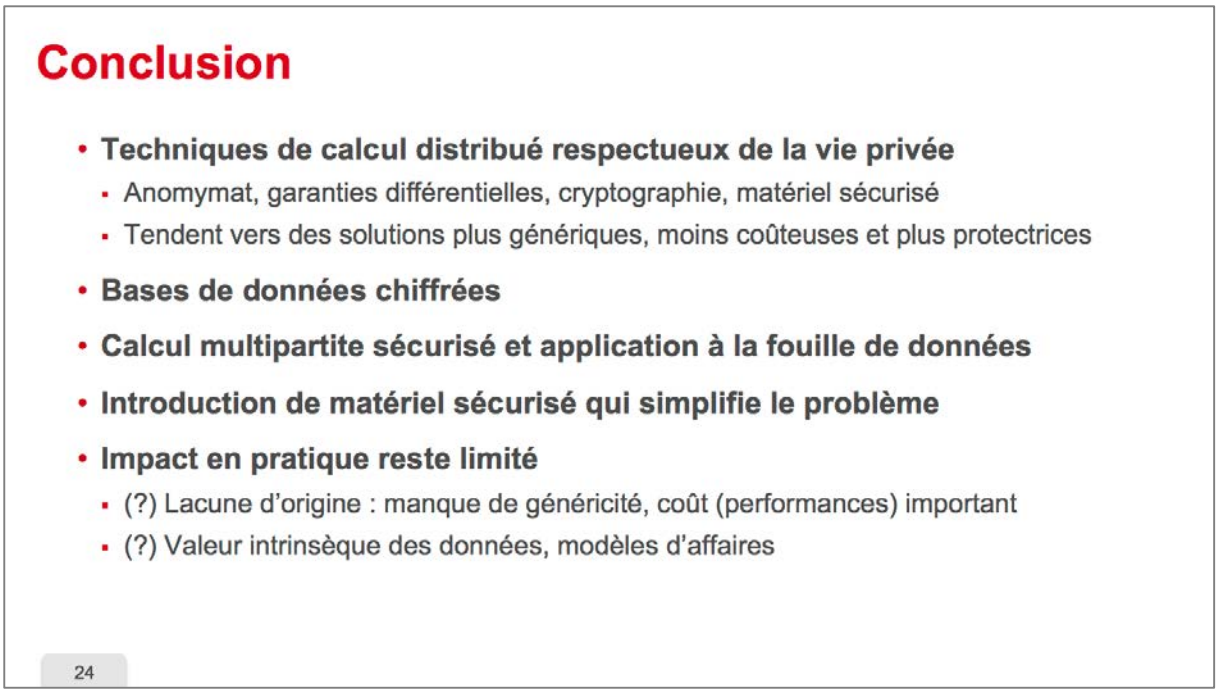
Le matériel sécurisé déchiffre la requête, évalue localement l'ensemble des données autorisées et pertinentes pour cette requête et transmet ces données chiffrées à l'infrastructure.



Les données sont partitionnées par l'infrastructure et chaque participant en récupère une partition.



Une phase de calcul peut commencer. Chaque dispositif sécurisé réalise un calcul partiel sur sa partition et renvoie les résultats chiffrés à l'infrastructure.
Un mécanisme itératif s'engage jusqu'à ce que le résultat final puisse être construit et le résultat final une fois construit peut être récupéré et déchiffré par l'individu ayant posé la question.



En conclusion, de nombreuses techniques existent actuellement pour calculer des résultats globaux dans le respect de la vie privée.
Ces techniques sont basées sur de l'**anonymat**, du **chiffrement**, du **calcul multipartite sécurisé** ou du **matériel sécurisé**.
L'impact de ces techniques en pratique reste encore limité :

- (?) Lacune d'origine : manque de généricité, coût (performances) important
- (?) Valeur intrinsèque des données, modèles d'affaires

Cependant, l'effort normatif actuel, notamment conduit en Europe avec la refonte de la directive de 95, et les effets de bord de l'affaire Snowden nous permettent d'espérer une accélération de l'utilisation de ce genre de techniques au bénéfice des villes intelligentes.

Références 1/2

- [Sweeney02] Sweeney, L. (2002). Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *10*(5).
- [Kifer07] Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(3).
- [Li07] Li, N., Li, T., & Venkatasubramanian, S. (2007, April). t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE 2007*.
- [Yao82] Yao, A.C.: Protocols for secure computations. In Annual Symposium on Foundations of Computer Science, FOCS, 1982.
- [Dwork06] Dwork, C. (2011). Differential privacy. In *Encyclopedia of Cryptography and Security*.
- [Bambauer14] Bambauer, J., Muralidhar, K., & Sarathy, R. (2014). Fool's Gold: an Illustrated Critique of Differential Privacy. Vand. J. Ent. & Tech. L., 16, 701.
- [Clifton02] Clifton, C., Kantarcioglu, M., Vaidya, J., Lin, X., & Zhu, M. Y. (2002). Tools for privacy preserving distributed data mining. *ACM Sigkdd Explorations Newsletter*, *4*(2).
- [To14] To, Q. C., Nguyen, B., & Pucheral, P. (2014). Privacy-Preserving Query Execution using a Decentralized Architecture and Tamper Resistant Hardware. In *EDBT*'14.

Références 2/2

- [Hacigums'02] Hacigümüş, H., Iyer, B., Li, C., & Mehrotra, S. (2002). Executing SQL over encrypted data in the database-service-provider model. *ACM SIGMOD*.
- [AdaPopa'11] Ada Popa, R., Redfield, C. M. S., Zeldovich, N. and Balakrishnan, H. (2011). CryptDB: Protecting Confidentiality with Encrypted Query Processing. *ACM SOSP*.
- [Mohammed10] Mohammed, N., Fung, B., Hung, P. C., & Lee, C. K. (2010). Centralized and distributed anonymization for high-dimensional healthcare data. *ACM TKDD, 4*(4).
- [IBM] IBM 4764 Model 001 PCI-X Cryptographic Coprocessor. Lien : https://www-03.ibm.com/security/cryptocards/pdfs/4764-001_PCIX_Data_Sheet.pdf
- [Bajaj14] Bajaj, S., & Sion, R. (2014). TrustedDB: A trusted hardware-based database with privacy and data confidentiality. In *IEEE TKDE*, *26*(3).
- [Javinen10] Jarvinen, K., Kolesnikov, V., Sadeghi A-R., Schneider, T.: Embedded SFE:Offloading Server and Net-work Using Hardware Tokens. In Financial Cryptography and Data Security (2010)
- [Katz07] Katz, J.:Universally Composable Multi-party Computation Using Tamper-Proof Hardware. In EUROCRYPT, 2007.
- [Allard10] Allard, T., Anciaux, N., Bouganim, L., Guo, Y., Le Folgoc, L., Nguyen, B., Pucheral, P. & Yin, S. (2010). Secure personal data servers: a vision paper. *In PVLDB*, *3*(1-2).